26-3-2012

MÉTODOS DE ELIMINACIÓN DE ARTEFACTOS Y CORRECCIÓN DE LA SEÑAL ESTEREOSCÓPICA.

Autor: Óscar Amatriaín Lapuerta

Coordinador: Jacek Dusza

Proyecto Fin de Carrera

Índice del contenido.

Índice de tablas.

Índice de ilustraciones.

1.- Introducción.

El presente proyecto está basado en la investigación del movimiento del cuerpo humano cuando este se encuentra andando. Dicho proyecto es llevado a cabo por la Universidad Politécnica de Varsovia, bajo la supervisión del profesor Jacek Dusza y la Universidad de Valencia.

Consta de la adquisición de datos correspondientes a 26 puntos (reflectantes que nos dan las coordenadas X,Y,Z) repartidos estratégicamente en el cuerpo para una vez realizado el tratamiento de señal correspondiente, juzgar el comportamiento de dicho cuerpo humano y poder observar si todo está correcto o el individuo puede tener diversas patologías.

El proceso comienza por la colocación de los reflectantes, los cuales han sido colocados, de tal manera que cubren todo el cuerpo del sujeto (siempre manteniendo dichas posiciones para todos los pacientes que sometemos a la prueba). A continuación y una vez visto que los reflectantes están colocados y preparados, el paciente ha de andar sobre una cinta. De este modo y utilizando cuatro cámaras de infrarrojos conseguimos obtener las coordenadas de los 26 puntos reflectantes en espacio y tiempo. Los datos serán guardados en un documento para más tarde poder tratarlos y llegar a los resultados que reflejarán las anomalías del paciente de una forma clara, rápida y precisa.

Una vez tenemos los datos pueden surgir problemas como las reflexiones, el no correcto funcionamiento de los dispositivos, la precisión de la representación de dichos datos…Todos estos problemas se traducen en que los datos obtenidos no reflejan la realidad y es por ello por lo que necesitamos tratarlos, para evitar falsos datos y errores.

En esto consistirá el presente proyecto, en filtrar los datos para quedarnos con los que sí que nos son útiles y poder a partir de ellos reconstruir el movimiento y el cuerpo del paciente.

El tratamiento de señal de este proyecto comenzó con otros proyectos como el llevado a cabo por el Dr. Jacek Dusza y Eduardo Daza Sánchez: “El Desarrollo de un conjunto de las funciones Doble Paso Medio según criterios de segmentación”.

Este proyecto hace uso del mencionado anteriormente, de tal manera que el objetivo principal es el tratamiento y corrección de los datos para la obtención de los resultados finales, útiles y aplicables para el proyecto “DPM” (Doble Paso Medio).

Summary.

This Project is based on research of human body when the subject's body performs a walking movement. This document is conducted by the Technical University of Warsaw, with Professor Jacek Dusza as coordinator and the University of Valencia.

It consists in data acquisition of 26 points (reflectors, which give us the coordinates X, Y and Z) strategically distributed in the body of our patient. Then we can study the behavior of the human body and observe if everything is correct or the patient has some pathology.

The process begins placing the reflectors, which have been distributed along the whole body of the subject, always keeping reflectors' positions for different patients. Once the reflectors are correctly positioned the patient has to walk on a treadmill. Four infrared cameras are used to obtain the coordinates of 26 reflectors in space and time. Data will be stored in a data basis so that later we can go through it and get the results to decide if the patient is healthy or not in a quick, accurate and easy way.

Considering the treatment of the data, problems appear with reflections, devices, accuracy of the representation of the data… All these problems show that data doesn't reflect reality and that is why we need to treat it to avoid false data and errors.

The aim of the project consists in filtering the data to achieve the clearest results so that later correct representation of the patient's body and his movement can be done.

The signal processing developed in this project began with other projects like the one carried out by Dr. Jacek Dusza and Eduardo Sánchez Daza: "The development of a set of functions as Double Middle Step segmentation criterion".

Contents of the previous project are used in this project, as the main target is treatment and correction of data to get useful and successful results for the "DMS" (Double Middle Step).

2.- Participantes en el proyecto.

El proyecto es llevado a cabo por Óscar Amatriaín Lapuerta, estudiante de la Universidad Pública de Navarra, bajo la supervisión del director del proyecto el Dr. Jacek Dusza.

La recogida de datos fue llevada a cabo por el departamento del Grupo SATI en Valencia.

3.- Objetivos del proyecto.

El cuerpo humano es una “máquina” hecha para andar. Es por ello que es muy importante que dicha acción sea realizada correctamente. Este es por tanto el principal objetivo del proyecto, observar si el paciente sufre alguna lesión o patología a la hora de estar en movimiento. Estamos tratando un proyecto de ingeniería aplicado a la medicina, con lo cual hemos de lograr que un médico pueda interpretar los resultados que nosotros le mostremos para determinar si el paciente está sano o si tiene algún problema físico.

Al andar utilizamos la mayoría de músculos y huesos de nuestro cuerpo y es por eso que colocaremos reflectantes en la cabeza, la columna, la cadera, los brazos y las piernas, cubriendo de esta manera los puntos más importantes del individuo para obtener resultados correctos y precisos.

Más concretamente los principales objetivos para llevar a cabo el proyecto serán:

-Eliminar las reflexiones producidas en la recogida de datos.

-Enderezar y fijar correctamente el cuerpo para que tenga una forma definida y precisa.

-Centrar el cuerpo en el sistema de coordenadas para la correcta visualización.

-Elaborar un programa que muestre el movimiento del cuerpo según el plano que queramos ver.

-Filtrar y tratar las señales para obtener la parte que es buena.

-Mezclar nuestra herramienta con la de otros proyectos anteriores para obtener el mejor resultado.

-Observar las diferencias y similitudes dependiendo del paciente y de sus señales.

-Recoger conclusiones.

Básicamente trataremos de crear una herramienta que tome los datos de entrada que tenemos, los trate eliminando los errores y guarde los datos corregidos. Para ello más adelante se verá que para llegar al objetivo podemos tomar varios caminos, dependiendo de la señal que tengamos y la forma de

tratarla.

El programa que utilizamos para crear dicha herramienta es “Matlab” (Matlab 7.8.0 R2009a). La herramienta está basada en una “GUI” (Graphical User Interface) para hacer más fácil, visual y manejable el tratamiento de las señales. De esta forma cualquiera podrá utilizar dicho programa sin falta de tener conocimientos avanzados de programación o simplemente como ya hemos dicho, está orientado a la medicina y cualquier médico podría de esta manera utilizarlo.

4.-Estructura del programa.

El presente proyecto comienza con los datos de los que disponemos, los datos de entrada, que reflejan las coordenadas X, Y, Z de los 26 puntos reflectantes colocados estratégicamente en el cuerpo del paciente. Los puntos reflectantes fueron grabados por cuatro cámaras de infrarrojos generando de esta forma la matriz de los datos en un documento con formato ".txt".

La colocación de los puntos reflectantes es la siguiente:

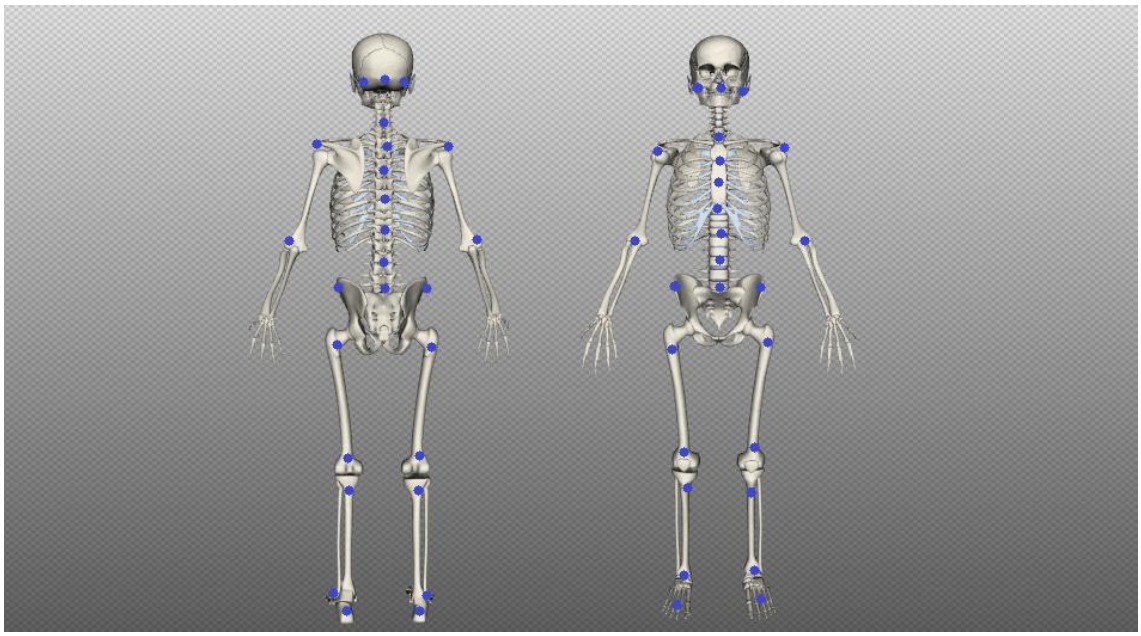


Ilustración 1. Colocación de los puntos reflectantes.

La recogida de datos se hizo para numerosos pacientes y con diferentes pesos por cada paciente que se sometía a la prueba. Es por eso que si vemos los datos, tendremos la carpeta con el nombre del paciente y luego tenemos diferentes carpetas cuyos nombres son los pesos que mantenían los pacientes mientras andaban sobre la cinta. Para que los archivos fueran más claros los nombres de los ficheros ".txt" se llaman 'modelo y nº'_'peso'_'cámaras'. Es decir, si por ejemplo tenemos el fichero "m5_06_13.txt" quiere decir que es el modelo 5, con peso 6 y se han utilizado las reconstrucciones de las cámaras 1 y 3.

En cuanto a los pesos, se cogieron medidas para varios pesos y la forma de colocar ese peso era mediante una correa que colgaba del hombro del paciente y de la cual colgaba a su vez la pesa correspondiente, quedando esta

debajo de la axila, como si de un bolso se tratara. De este modo conseguimos colocar el peso correspondiente al paciente sin tapar ningún reflectante ya que sino no podríamos grabar los datos de dicho punto.

Los ficheros contienen 450 filas y 79 columnas cada uno, lo que significan 450 muestras por cada coordenada de cada punto reflectante de modo que tenemos 450 muestras para la coordenada X del punto reflectante 1, 450 para la coordenada Y del punto 1, 450 para la coordenada Z del punto 1, 450 para la coordenada X del punto 2 y así sucesivamente para los 26 puntos reflectantes. La última columna es una escala de tiempos que utilizaremos para en nuestra herramienta dibujar el modelo y variar la velocidad de movimiento del mismo.

En la siguiente tabla podemos apreciar lo comentado más claramente:

Punto 1			**Punto 2**			**……**	**Punto 26**			
X	Y	Z	X	Y	Z	……	X	Y	Z	**ΔT**
X_{1-1}	Y_{1-1}	Z_{1-1}	X_{2-1}	Y_{2-1}	Z_{2-1}	……	X_{26-1}	Y_{26-1}	Z_{26-1}	**ΔT**
X_{1-2}	Y_{1-2}	Z_{1-2}	X_{2-2}	Y_{2-2}	Z_{2-2}	……	X_{26-2}	Y_{26-2}	Z_{26-2}	**ΔT**
……	……	…..	……	……	…..	……	……	……	…..	**……**
……	…..	…..	……	…..	…..	……	……	……	……	**……**
……	…..	…..	……	…..	…..	……	……	……	……	**……**
X_{1-450}	**Y_{1-450}**	**Z_{1-450}**	**X_{2-450}**	**Y_{2-450}**	**Z_{2-450}**	……	**X_{26-450}**	**Y_{26-450}**	**Z_{26-450}**	**ΔT**

Tabla 2. Tabla de datos de entrada.

En definitiva nuestra matriz constará de 3 columnas de coordenadas por cada sensor, siendo éstos 26 y 450 muestras por cada coordenada, obteniendo así 26*3=78 columnas, más la columna de escala temporal. Al final obtenemos una matriz de 450 filas y 79 columnas.

Como se puede ver en el siguiente dibujo, la coordenada X representa el movimiento lateral (de izquierda a derecha, left-right), la coordenada Y representa el movimiento vertical (de arriba abajo, up-down) y la coordenada Z representa el movimiento frontal (de adelante a atrás, forward-back).

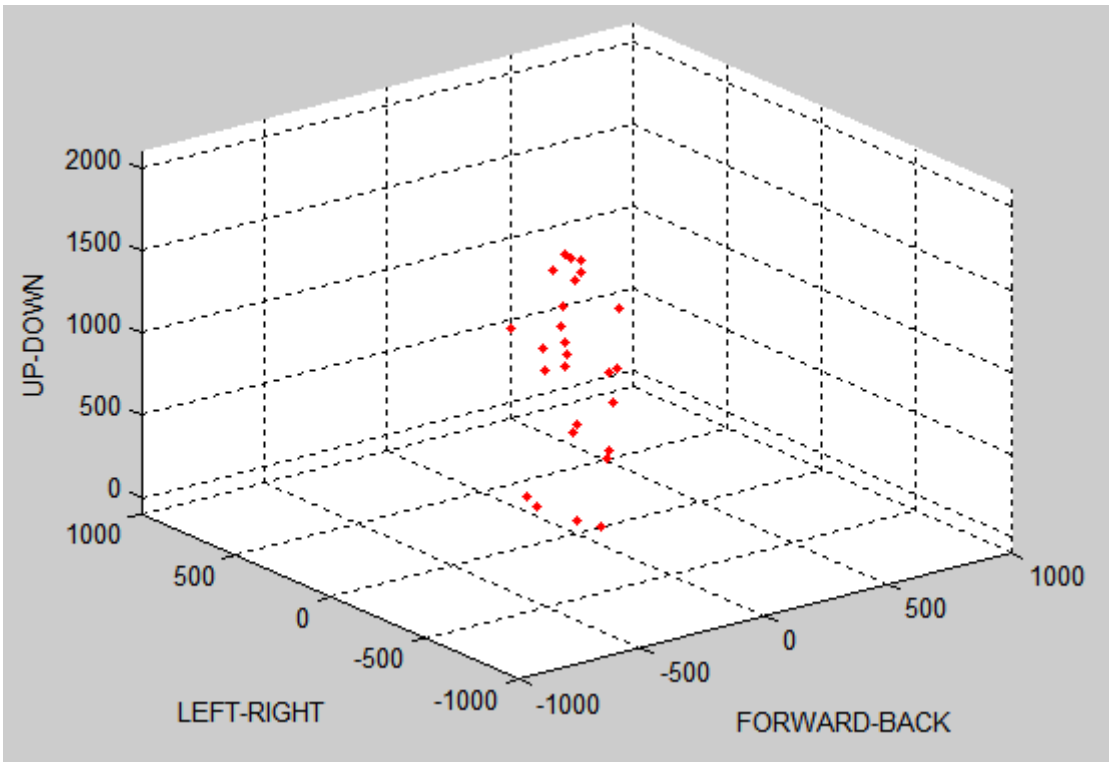


Ilustración 2. Coordenadas X, Y y Z.

La estructura de los datos para todos los pesos y pacientes es la misma y fue recogida en el laboratorio del grupo SATI en Valencia tal y como explicaremos a continuación.

En primer lugar, para la grabación de los datos se necesitaron cuatro cámaras de infrarrojos, 26 reflectantes para el movimiento, una cinta de andar y la sala acondicionada con materiales anti reflejantes (para evitar posibles errores de reflexiones) donde se hizo la recogida de datos. El laboratorio consistía en lo siguiente:

1. Cinta de andar: Se trata de una máquina donde el paciente ha de subirse para empezar a grabar los datos ya que si no tuviéramos dicho elemento el paciente debería desplazarse en el espacio y sería mucho más complicada tal recogida.
2. Reflectantes: 26 dispositivos reflectantes son los encargados de que las cámaras puedan recoger las coordenadas del paciente. Como hemos visto anteriormente están colocados de tal manera que podamos reconstruir más tarde el cuerpo del paciente lo mejor posible.
3. Cámaras: 4 cámaras se utilizan para recoger la posición de cada punto reflectante en espacio y tiempo. Estas cámaras son situadas en las cuatro esquinas tal y como se ve en el dibujo, de manera que utilizando

infrarrojos pueden detectar las reflexiones de los reflectantes.

4. Ordenador: en él se guardan los datos recogidos durante la sesión en el formato explicado anteriormente.

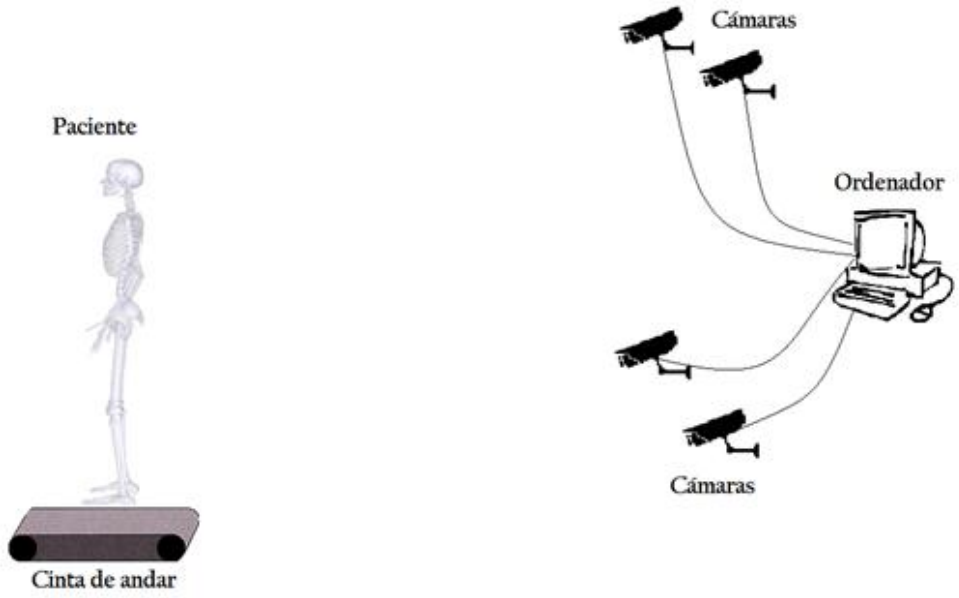


Ilustración 3. Estructura del laboratorio.

El método consistía en que el paciente tuviera, mientras caminaba sobre la cinta, 26 reflectantes colocados estratégicamente de manera que las cuatro cámaras recogieran las señales de dichos reflectantes. De esta forma, en tiempo y espacio obtenemos las coordenadas necesarias para luego poder representarlas en el ordenador.

Al tener cuatro cámaras podríamos tener hasta 6 combinaciones posibles utilizando 2 cámaras por combinación, pero los datos se han tomado según cuatro combinaciones. Con dichas combinaciones somos capaces de hacer las reconstrucciones 3 D y así obtener los ".txt" donde tenemos los datos preparados para el programa. Sin embargo tenemos diferentes errores según el punto reflectante en el que nos fijemos. Es decir, los 3 puntos de la cabeza y de la columna son grabados por todas las cámaras, tenemos redundancia de datos, y al hacer la reconstrucción tenemos un error menor a 0,6 mm, sin embargo el resto de puntos son grabados por algunas cámaras pero quedan escondidos para otras, por lo que el error es mayor, pero menor de 1,6 mm.

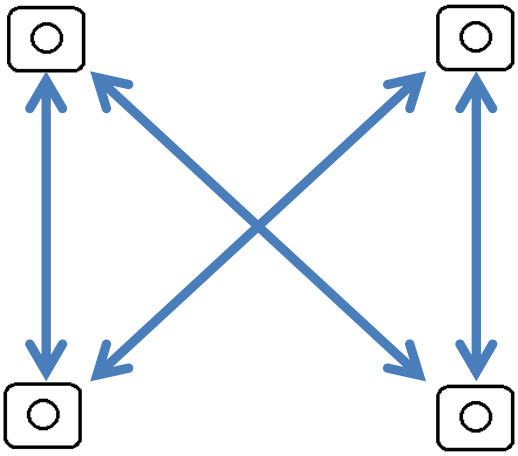


Ilustración 4. Combinaciones de las cámaras.

El resultado sería una señal tal y como muestra la siguiente imagen.

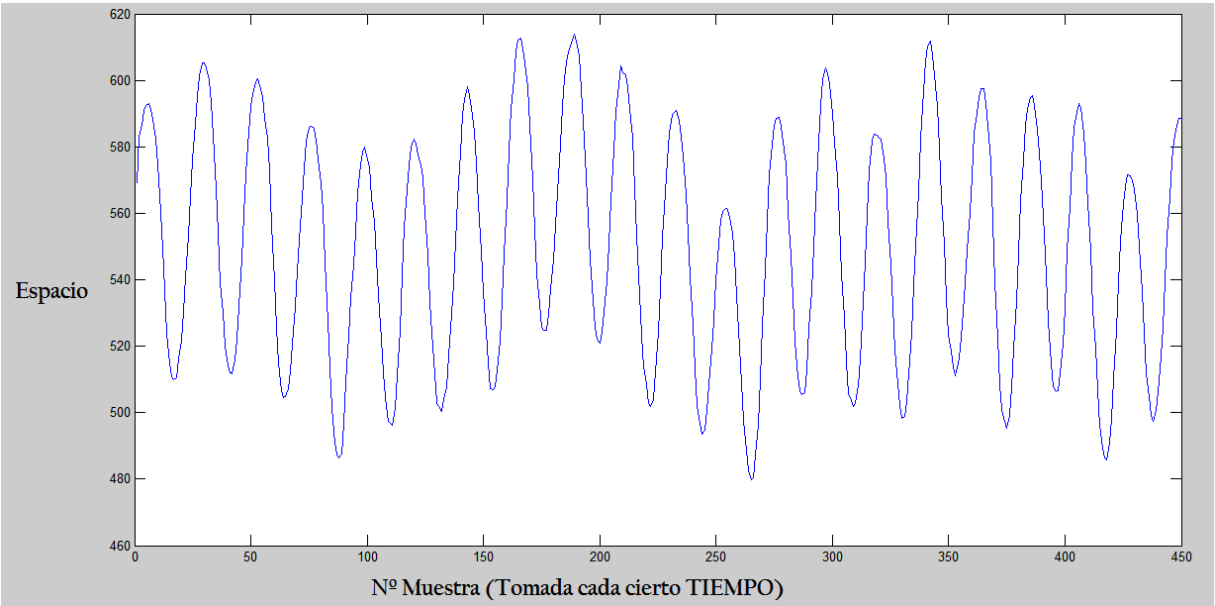


Ilustración 5. Ejemplo de señal tomada en el laboratorio.

Sin embargo, como los resultados no fueron correctos en todas las medidas y se obtuvieron señales incorrectas debido a los errores por reflexión, hemos de tratar señales como la que se ve a continuación, donde se ven claramente picos que demuestran que se han producido fallos en el sistema a la hora de medir.

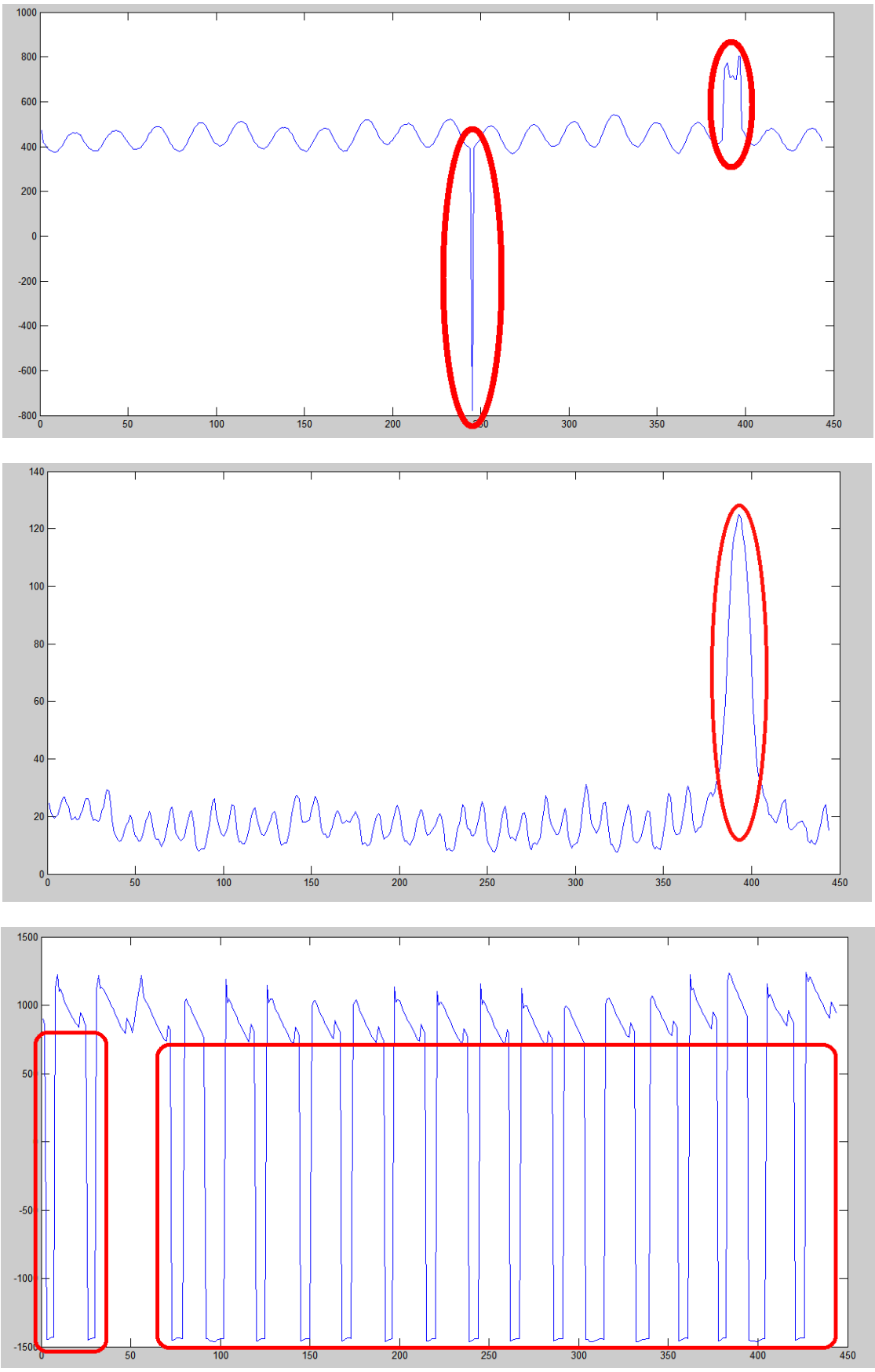


Ilustración 6. Ejemplos de señales con errores en la recogida de datos.

El programa ha sido dividido en varios subprogramas para hacer que sea más fácil el entenderlo y para reducir carga a la hora de ejecutarlo ya que así sólo ejecutamos los subprogramas que queremos usar. Por otro lado como hemos comentado en un apartado anterior, hemos utilizado interfaces gráficas (GUI) para ver y tratar las señales de una manera más sencilla.

En definitiva, han sido creados dos programas, diferenciados por la funcionalidad de cada uno de ellos. El primero, “Program”, es el que utilizamos para tratar y corregir las señales y sin embargo el segundo, “Ver_DPM”, se utiliza para visualizar los resultados finales, sin opción de modificar nada.

Con respecto a la “GUI”, tendremos un fichero “.fig” y otro fichero “.m”. El fichero “.fig” es el encargado de ejecutar la interfaz gráfica y el fichero “.m” es el que tiene el código necesario para que todo funcione correctamente, la interfaz y nuestro programa.

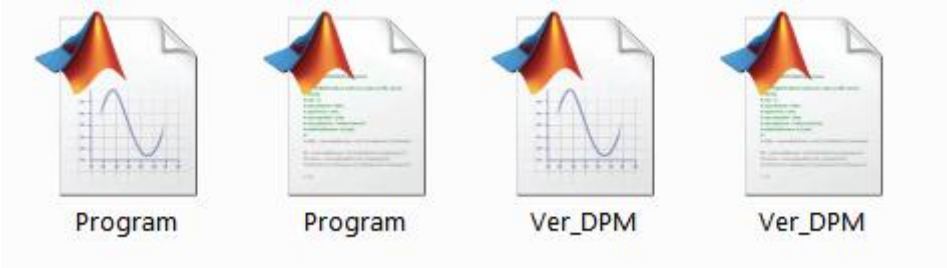


Ilustración 7. Ficheros ".fig" y ".m" de ambos programas.

El contenido de nuestra carpeta (Nombrada “Programa”) donde estará nuestro programa tendrá los siguientes archivos:

Ilustración 8. Aspecto de la carpeta, "Programa".

Como podemos observar, los 2 ficheros cuyo icono es una gráfica y están seleccionados son los que corresponden a los archivos “.fig” que abrirán nuestra interfaz gráfica.

En cuanto al resto de contenidos, la carpeta “Datos” contiene el fichero “_Modelo_.txt”, necesario para que funcione el programa y además contiene a su vez carpetas con el nombre de los pacientes y dentro de ellas otras con

nombre 0, 2, 4, 6, 8, 10 y 15, las cuales indican el peso que llevaba el paciente durante la toma de datos. Dentro de cada una de ellas están las reconstrucciones con los archivos ".txt".

En la carpeta "Datos Corregidos DPM" se almacenan los resultados finales después de haber sido tratados y en "Datos Corregidos" se guardan los mismos datos antes de aplicar el "DPM".

Por último y más importante, los programas y archivos temporales utilizados para el tratamiento de las señales están en la carpeta "data". Esta carpeta contiene los siguientes programas, que serán explicados en los siguientes sub-apartados:

Ilustración 9. Subprogramas en la carpeta "data".

NOTA: Dichas carpetas y archivos (incluido "_Modelo_.txt") han de estar siempre tal cual se ha explicado para el correcto funcionamiento de los programas. Si se quisiera poner los datos de otro paciente, bastaría con ubicar dichos datos en la carpeta "Datos".

0.- Esquema básico del programa.

En el siguiente esquema se representan, para hacer más fácil el entendimiento del programa, las llamadas y el funcionamiento entre el programa principal y el resto de programas una vez pulsado el botón “Ejecutar Programa” en la interfaz gráfica “Program.fig”.

Dicho botón es el que se encarga de ejecutar la mayor parte del código que trata las señales y elimina los artefactos, es por ello que se hace más hincapié en él y no el del resto de botones, pues el resto ejecutan códigos más sencillos y sin llamamiento a otros subprogramas.

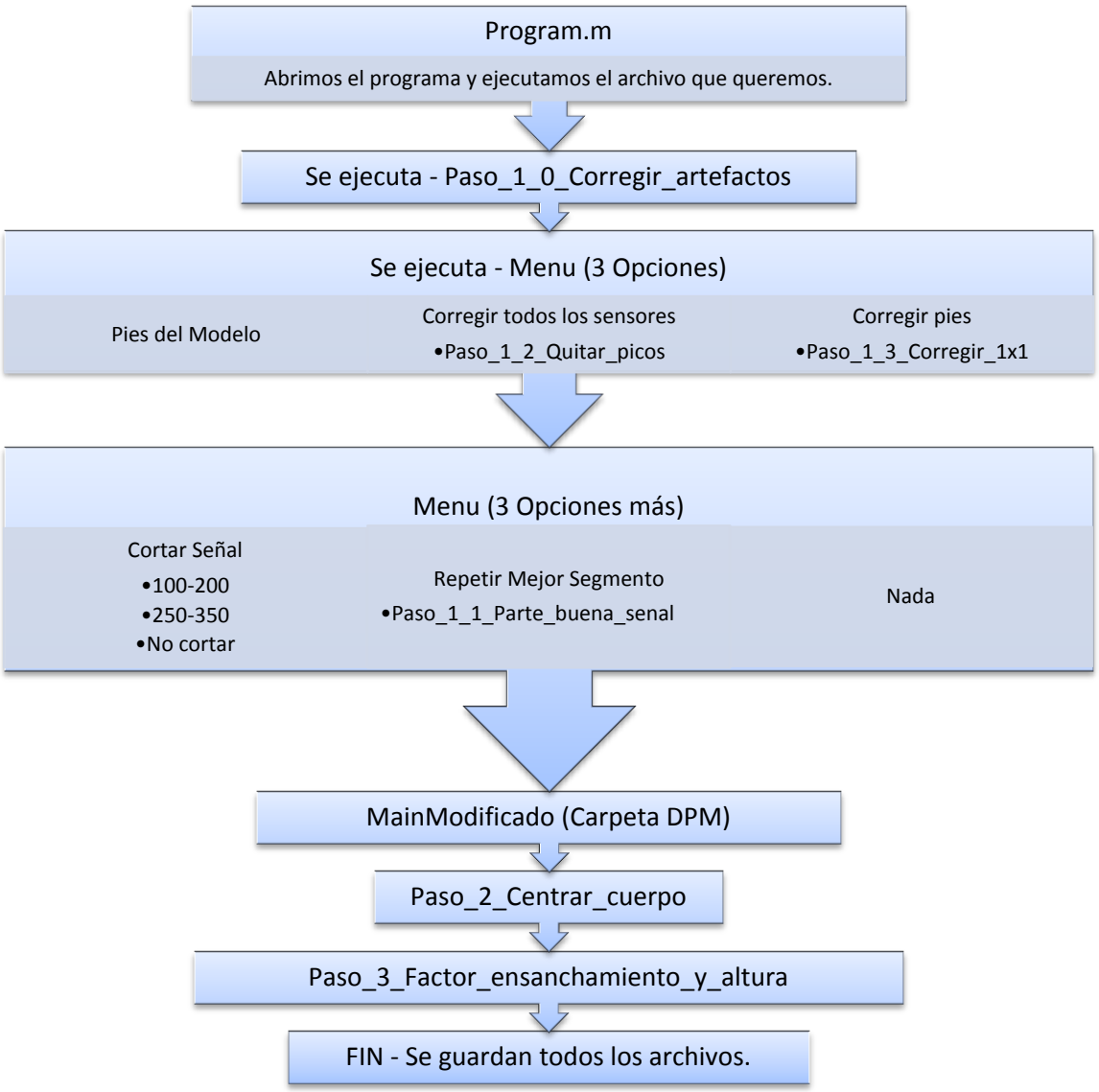


Ilustración 10. Esquema del programa y subprogramas.

No está incluido dentro del organigrama anterior el subprograma “Columna” debido a que este archivo es utilizado en el programa “Ver_DPM” para analizar la posición de la columna dependiendo del peso que el paciente lleve colocado.

Lo que refleja el organigrama es básicamente lo que lleva a cabo el programa “Program.m”, que es en definitiva el código donde se llama a los demás subprogramas, el “índice” de todo el trabajo que se realiza.

Por otra parte también está incluido en dicho archivo (Program.m) todo el código de los demás botones.

Ver código “Program.m”

1.- Paso_1_0_Corregir_artefactos.

El principal objetivo de nuestro proyecto es la corrección de las secciones defectuosas de las señales de las que disponemos. Es por esto que hemos creado este subprograma, el cual se encarga de corregir los diferentes problemas que tenemos en nuestras señales. Para comprender bien el código será necesario explicar previamente el sistema o estructura que hemos seguido.

En primer lugar y tras observar cuantiosas señales, hemos podido ver que no siempre los errores son iguales o son errores sistemáticos, sino que tenemos diferentes tipos de errores y por eso será necesario tratarlos de forma diferente. Los tipos de artefactos o errores se explican a continuación:

- Artefacto corto.

Nos referimos con ello a reflexiones producidas en muestras aisladas, es decir, que en una determinada muestra se produce una reflexión, por lo que ese valor ya no será correcto.

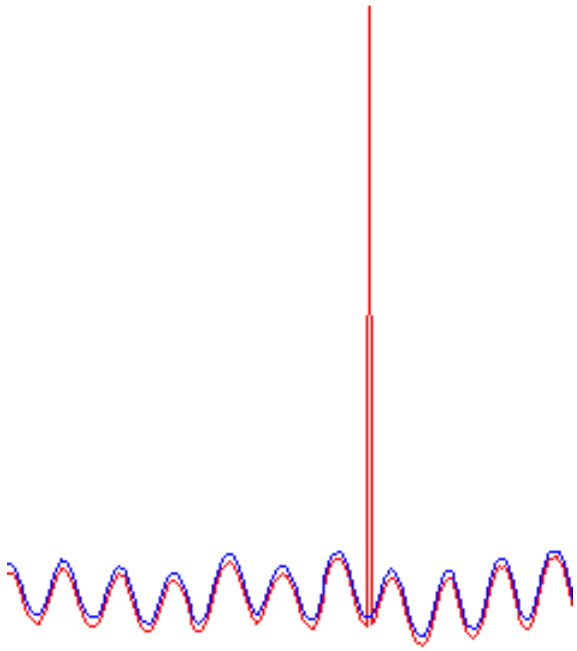


Ilustración 11. Artefacto corto.

La manera de tratar dicho error será igualando la muestra donde se ha producido el error al valor de la muestra anterior.

- Artefacto largo.

Se produce cuando ya no es sólo en una muestra donde tenemos errores sino que ha sido una cantidad de muestras consecutivas.

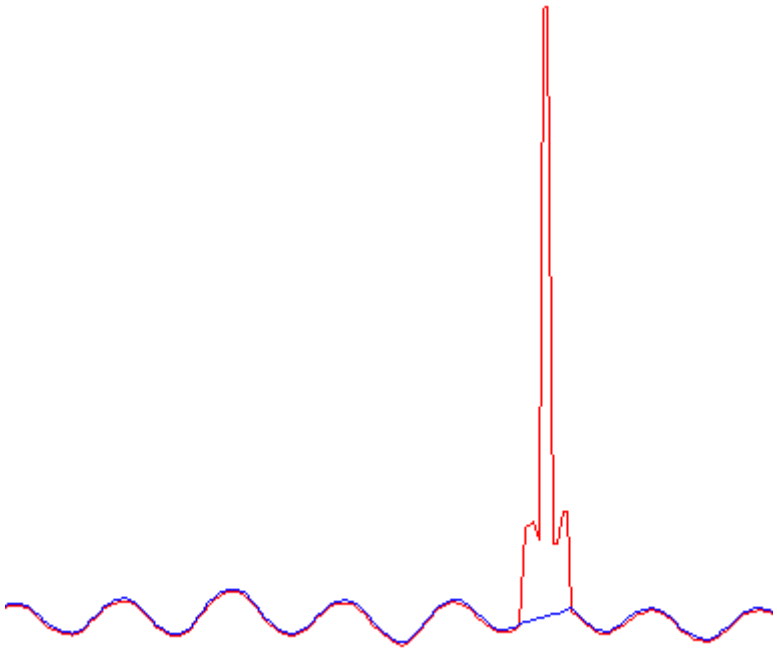


Ilustración 12. Artefacto largo.

La solución aplicada en este tipo de artefactos es detectar el flanco donde empiezan las muestras no correctas y aplicar la interpolación lineal entre la anterior muestra y posterior muestra del artefacto.

- Reflexiones continuas que se producen cada “x” muestras.

Podemos ver señales, típicamente las de los pies, donde los rangos de nuestras señales son muy grandes comparándolo con lo que en realidad deberían ser, es por ello que podemos ver que la mayor parte de la señal está tomada correctamente o sigue su forma natural pero está desplazada hacia arriba o hacia abajo. En el código, dependiendo del desplazamiento, se tratará la señal de una u otra manera.

En el siguiente dibujo podemos ver las partes buenas de la señal y los desplazamientos sufridos por dichas reflexiones:

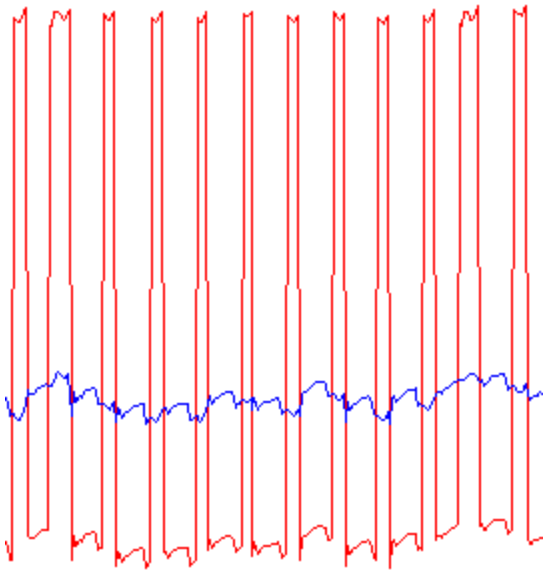


Ilustración 13. Reflexiones continuas que se producen cada "x" muestras.

En este caso, el algoritmo aplicado corrige dicha señal igualando la primera muestra en la que se ha producido el error al valor de la muestra anterior y restando a las siguientes muestras que no son correctas dicho rango, de modo que se copie la señal. En definitiva lo que estamos haciendo es desplazar la señal a su posición correcta.

- Señales irrecuperables.

Puede darse el caso de que haya señales que aun siendo corregidas sigamos teniendo problemas ya que por fallos en el sistema o reflexiones continuas los valores tomados no corresponden de ninguna manera a la realidad.

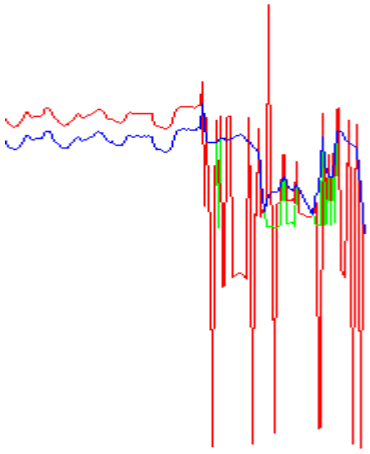


Ilustración 14. Señales irrecuperables.

El resultado de estas señales después de haberlas tratado e intentado corregir sigue no siendo el correcto y es por eso que decimos que son irrecuperables. La solución a esto es cortar la señal por la muestra correspondiente. Por eso al principio, cuando cargamos un archivo nos da la opción de cortar o de repetir el mejor segmento.

Este subprograma también corrige la posición de los puntos. Puede haber casos en los que una señal es entera desplazada, pudiendo aparecer en la representación un punto desviado que no refleja la correcta coordenada del paciente. El proceso es tomar como referencia las coordenadas del modelo y después de haber corregido los errores de las señales centrarlo en dicho rango. De esta manera nos aseguramos de que lo que estamos visualizando corresponde al paciente y a su vez está centrado mostrando la verdadera realidad.

Además, también puede haber fallos que provoquen que el cuerpo no esté completamente enderezado quedando por ejemplo la cadera adelantada respecto a la columna, la columna adelantada respecto a los puntos de la cabeza… Para ello también hay parte del código con el objetivo de ajustar estos valores y realizar una buena representación.

Ver código “Paso_1_0_Corregir_artefactos”

2.- Menú.

Este subprograma está basado en un listado de acciones que el usuario puede utilizar dependiendo de si cree que es o no conveniente.

Como su nombre indica, se trata de un menú, que se ejecuta automáticamente después de "Paso_1_0_Corregir_artefactos" mostrando las siguientes opciones:

Pregunta 1 - ¿Corregir la señal?

Opciones:

"Si" – Si elegimos esta opción nos muestra las siguientes opciones:

- "Pies del modelo" – Sustituimos los pies de nuestro paciente por los del modelo. Es aconsejable sobre todo cuando las señales de los pies contienen muchos errores, para así poder ver bien el resto del cuerpo.

- "Corregir todos los sensores" – Se aplica un algoritmo como en el paso anterior para seguir corrigiendo posibles artefactos.

- "Corregir pies" – Se muestran de una en una en una gráfica las señales correspondientes a las 3 coordenadas de cada uno de los 4 puntos de los dos pies antes y después de la corrección efectuada y aparece un cuadro por si queremos sustituir la señal o no.

"No" – Continua hacia el siguiente paso.

Pregunta 2 - ¿Cortar la señal?

Se nos muestra una gráfica con todas las señales y si se ven señales malas podremos cortarlas por la muestra que indiquemos.

Opciones:

"Si" – Si decidimos cortar la señal, podremos quedarnos desde la muestra 0 hasta la muestra:

- Opción 1 - 100-200

Una vez aquí volverá a abrirse un cuadro con la opción final:

-100

-150

-200

- Opción 2 – 250 – 350

Del mismo modo que en lo anterior:

-250

-300

-350

- Opción 3 – No cortar

Cancela el proceso de corte.

"No" - Seleccionando esta opción la imagen quedará intacta.

Para hacernos una idea clara del esquema del menú, podemos ver en la siguiente ilustración todas las opciones:

Al pulsar en los botones con el círculo rojo aparecerá lo que muestra la flecha, mientras que si pulsamos cualquier botón sin dicho círculo el programa continuará.

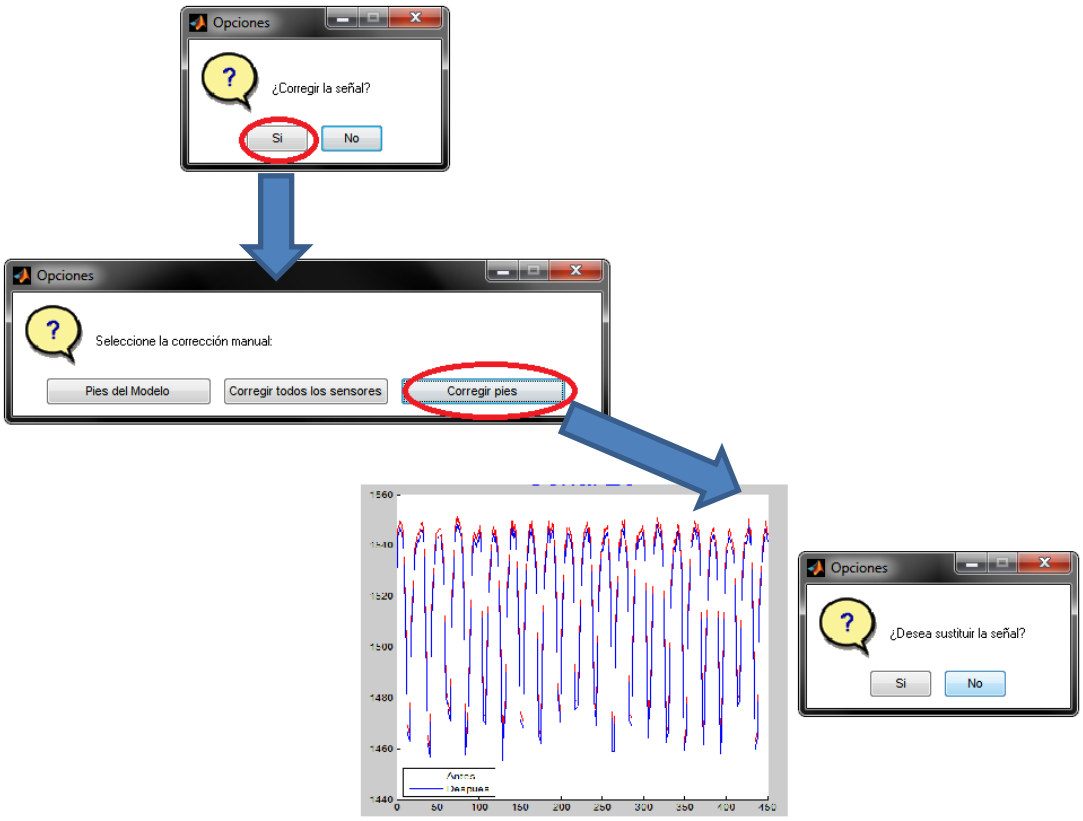
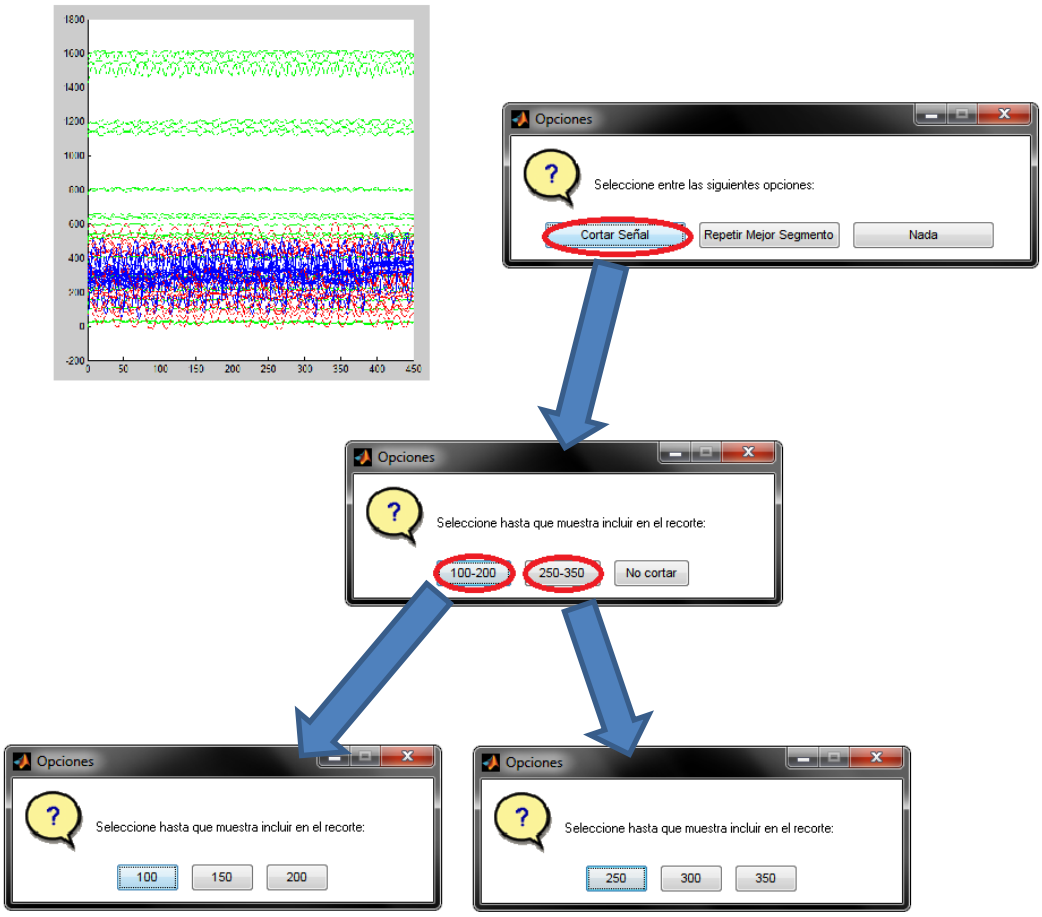


Ilustración 15. Esquema básico del subprograma "Menú".

Tanto si queremos corregir la señal como si no, a continuación seguirá con lo siguiente:



Ver código “Menú”.

3.- Paso_1_1_Parte_buena_senal.

Forma parte de una de las aplicaciones contenidas en “Menu” y aparece con el nombre “Repetir Mejor Segmento”. Actúa partiendo la señal en cuatro segmentos de 100 muestras cada una, a partir de las cuales calcula la desviación tìpica y se queda con la de menor. De esta forma conseguimos coger la mejor parte de la señal. En las siguientes ilustraciones podemos observar cómo se hace:

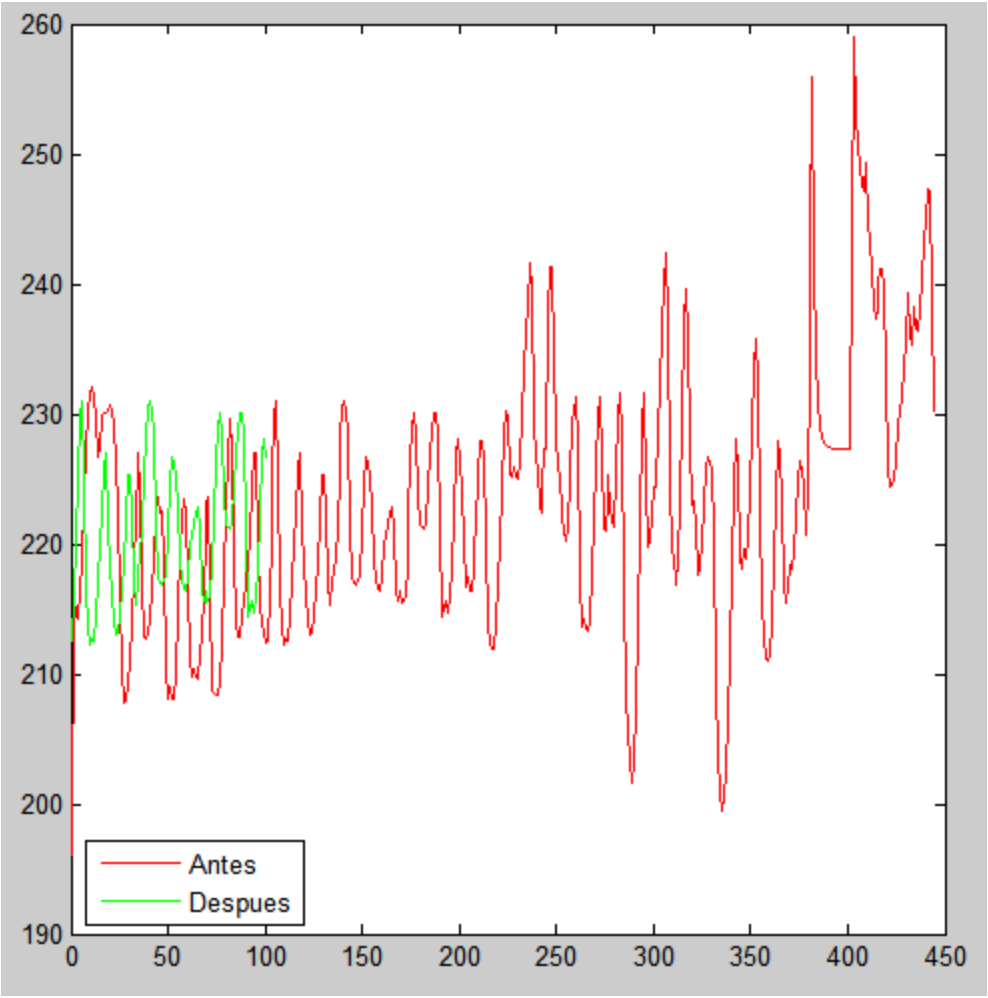


Ilustración 16. Ejemplo 1 Repetir mejor segmento.

En el ejemplo de arriba podemos ver como la parte de señal es la que menos picos y artefactos puede tener y en este caso ha recortado la señal quedándose con las muestras que van desde la 101 hasta la 200, tratándose del segundo segmento. Sin embargo en el siguiente ejemplo se ve que el segmento de menor desviación típica es el tercero, de la muestra 201 a la 300.

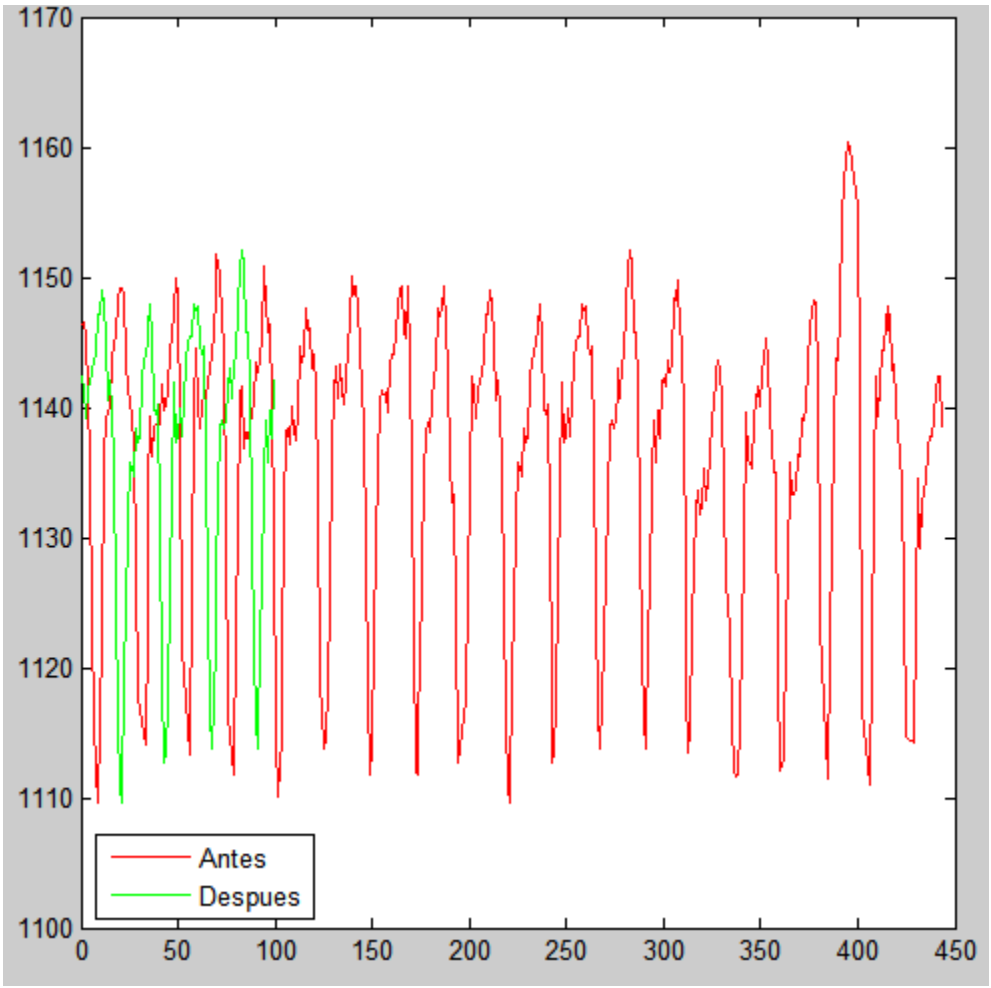


Ilustración 17. Ejemplo 2 Repetir mejor segmento.

Una vez seleccionado el mejor segmento, se lleva a cabo una concatenación del mismo para poder continuar con los siguientes programas sin ningún problema. De esta forma nuestra señal pasará a ser el segmento escogido repetido periódicamente.

Ver código "Paso_1_1_Parte_buena_senal".

4.- Paso_1_2_Quitar_picos.

Es otra de las aplicaciones contenidas en “Menu” y aparece con el nombre “Corregir todos los sensores”. El código de este subprograma se encarga de repasar todas las señales de todos los puntos reflectantes en busca de artefactos o reflexiones para eliminarlos. Una vez llamado este programa tratará las 78 señales automáticamente.

Aparece el nombre de sensor aunque en realidad no lo son ya que se tratan de reflectantes, pero en este caso se ha utilizado este término por hacer más sencilla la comprensión del programa.

Ver código “Paso_1_2_Quitar_picos”.

5.- Paso_1_3_Corregir_1x1.

Es una aplicación de “Menu” bajo el nombre de “Corregir pies” similar al anterior programa comentado arriba. Básicamente se trata de lo mismo, corregir las señales pero sin embargo este programa se ha centrado en los pies ya que son esos 4 puntos donde más problemas solemos tener. De esta forma al llamar a este programa se abrirá en una ventana la señal antes y después de ser corregida y un cuadro con el cual podemos decidir si queremos sustituirlas.

Ver código “Paso_1_3_Corregir_1x1”.

6.- Paso_2_Centrar_cuerpo.

Una vez tratadas todas las señales podemos verlas en nuestro gráfico en 3 dimensiones o en el plano que deseemos visualizar pero antes de ello hemos de centrarlo en el punto 0 para así observar los resultados de una manera simple y precisa. De esta forma, aplicando este algoritmo conseguimos que sea el paciente que sea, habrá sido colocado en el centro de todas nuestras gráficas y representaciones.

Por un lado en cuanto a la altura del modelo, se ha tomado como referencia el punto medio entre los dos pies. Dicho punto ha sido colocado en el

punto 0 de nuestras gráficas. Lo conseguido con ello es que con un simple vistazo a nuestras representaciones ya tenemos centrado nuestro paciente y podemos medir su altura con ayuda de la cuadrícula y los ejes.

Por otro lado, para centrarlo lateralmente, se han tomado como referencia todos los puntos de la columna, se ha hallado la media y de ahí se ha trasladado al punto 0. De esta forma nuestro paciente también quedará centrado en dicha coordenada, quedando la columna en el centro de nuestra gráfica.

En conclusión, con dichas correcciones es suficiente para una correcta colocación de nuestro paciente en el eje 0 de nuestras gráficas.

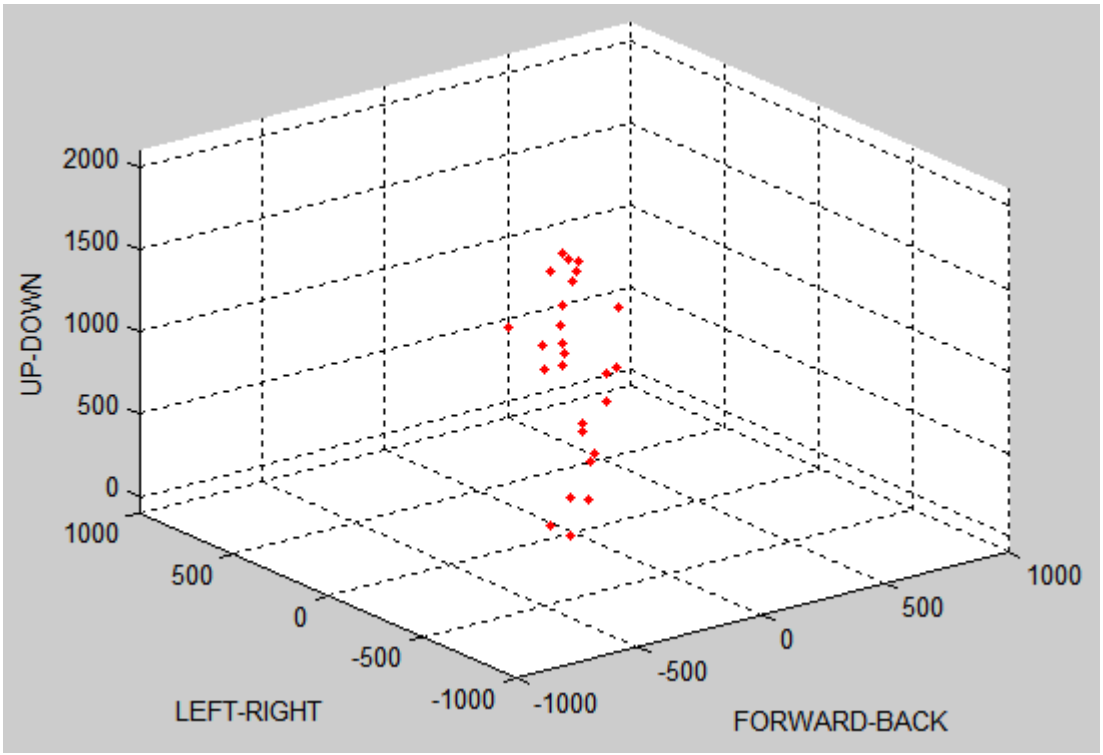


Ilustración 18. Centrar cuerpo - Representación 3D.

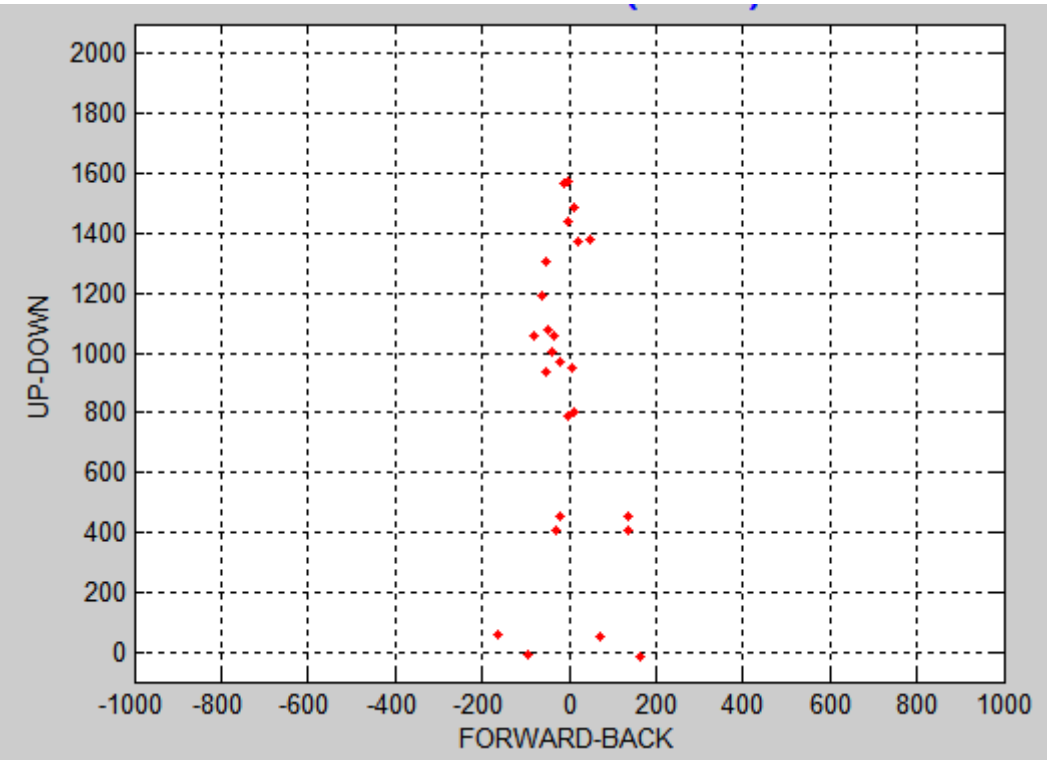


Ilustración 19. Centrar cuerpo - Plano Y-Z.

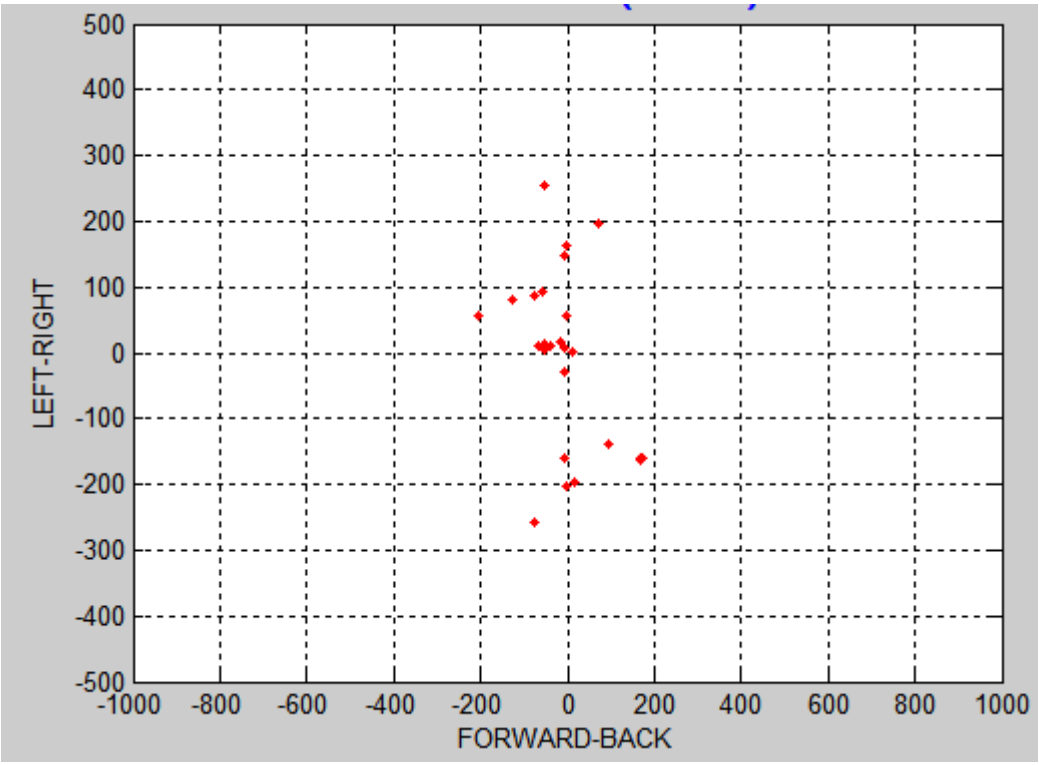


Ilustración 20. Centrar cuerpo - Plano X-Z.

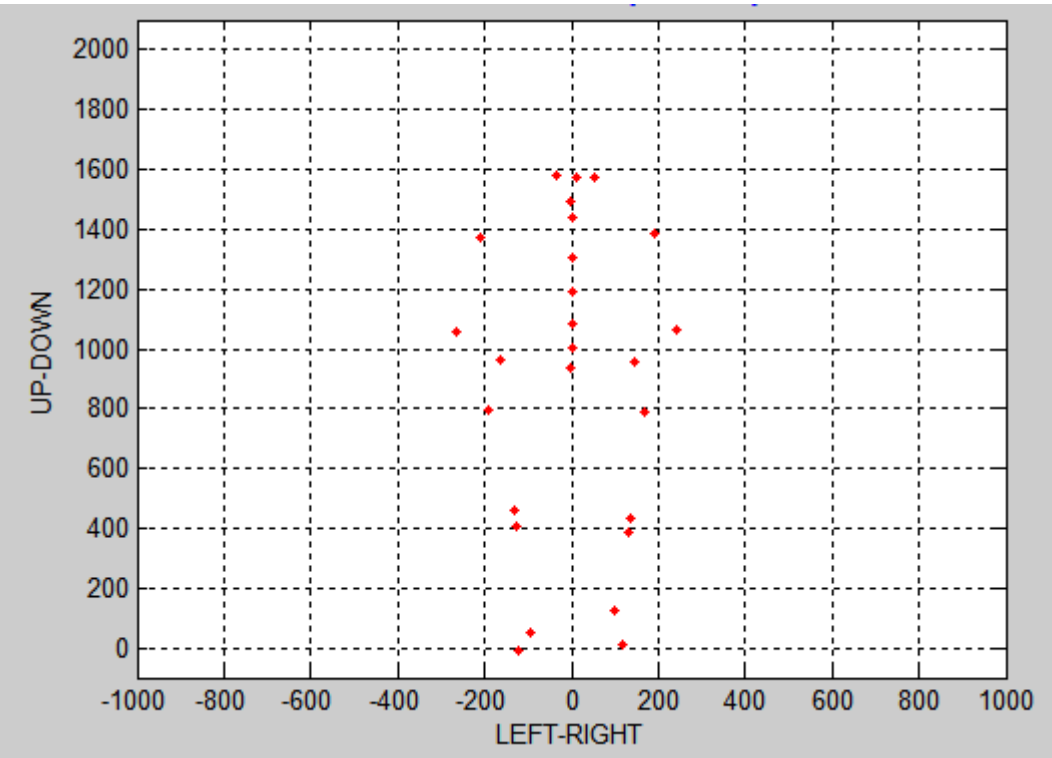


Ilustración 21. Centrar cuerpo - Plano Y-X.

Ver código "Paso_2_Centrar_cuerpo".

7.- Paso_3_Factor_ensanchamiento_y_altura.

En todo el programa estamos utilizando de referencia un modelo que es el que nos da los rangos y medidas correctos, pero no todos los pacientes tienen la misma altura o la misma anchura. Para solucionar esto se ha creado este programa.

El objetivo de este programa es sencillo, se trata de ensanchar o encoger nuestro modelo para que tenga las medidas correctas de nuestro paciente real. Para ello se comparan medidas de nuestro modelo y las del paciente y de aquí se sacan dos factores, el factor de la altura y el de anchura. Una vez obtenidos, se centra el punto medio entre la cadera de nuestro modelo en el cero. Así podemos aplicar los factores de multiplicación correctamente. Más tarde se traslada de nuevo para centrarlo como en el apartado anterior y dejarlo preparado para la visualización.

Para ver que dicho código funciona, ha sido creado un documento con las medidas no reales de un paciente el doble de grandes. Podemos ver como si que el programa nos modifica los valores dándonos una clara y real visión de nuestro paciente. En la figura vemos en color rojo la figura estática del modelo y en azul el resultado después de haberlo centrado y multiplicado por los factores correspondientes.

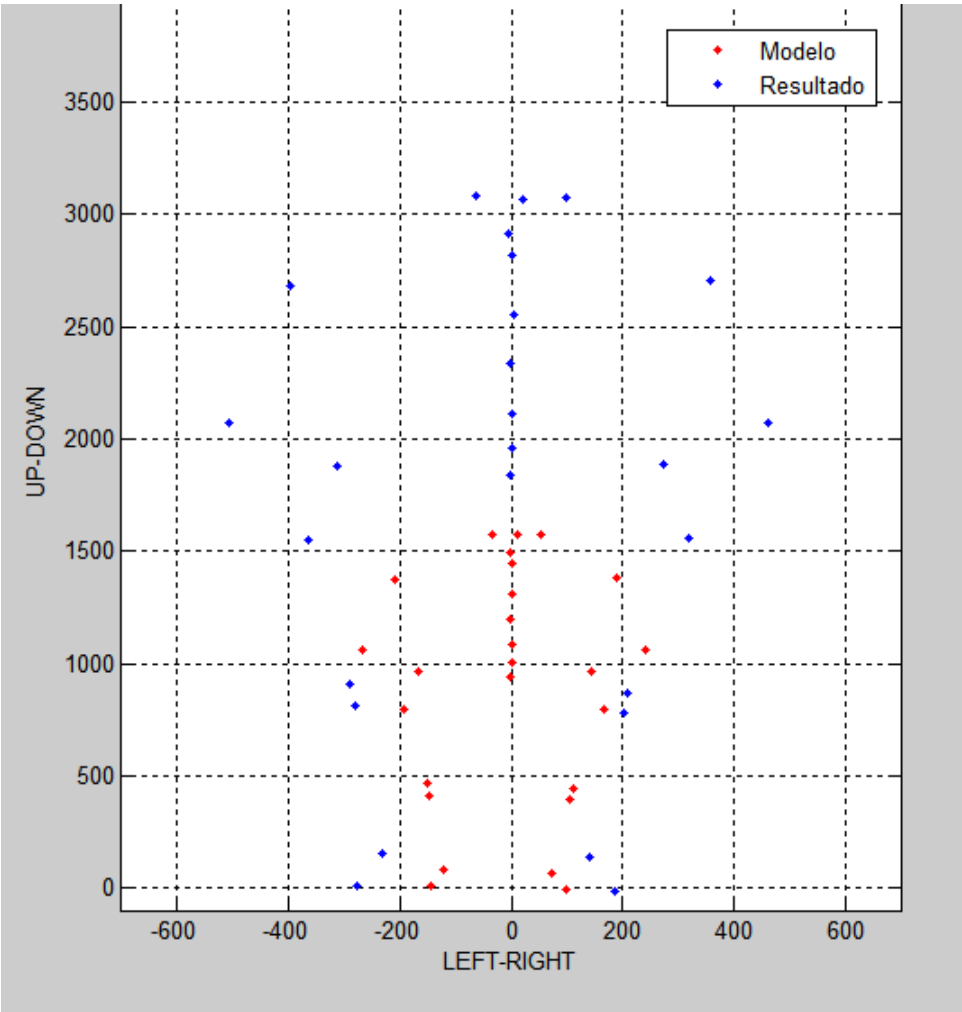


Ilustración 22. Ejemplo de Paso_3_Factor_ensanchamiento_y_altura.

Ver código "Paso_3_Factor_ensanchamiento_y_altura".

8.- Dibujar.

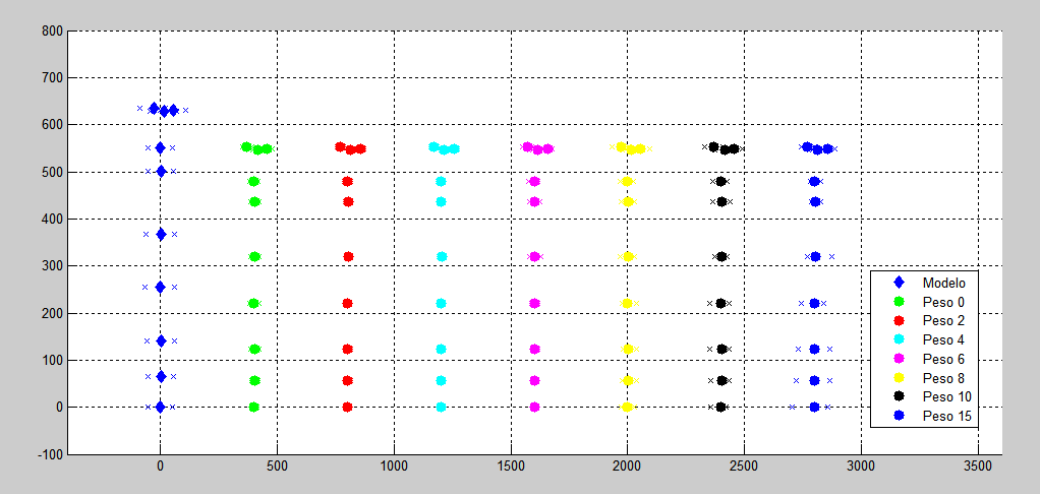
Este programa tiene el objetivo de representar los resultados. Está separado y no incluido en los demás programas porque es llamado por varios de ellos y de esta forma se reduce el tamaño del código y está más ordenado.

Si se ve el código se puede ver que cambiando un simple valor (la x) podemos representar los valores en una gráfica superpuestos o en varias gráficas.

Ver código "Dibujar".

9.- Columna

Este código está basado en la comparación de la columna vertebral cuando el paciente está sometido a no llevar carga, a llevar una carga de 2 kg, 4 kg, 6 kg, 8kg, 10kg y 15 kg. De tal modo que se ve en una gráfica la diferencia entre todos los casos mencionados, lo cual resulta muy útil para que un médico pueda obtener resultados y análisis sobre nuestro paciente. Además se pueden observar los rangos de movimiento de la columna de nuestro paciente.



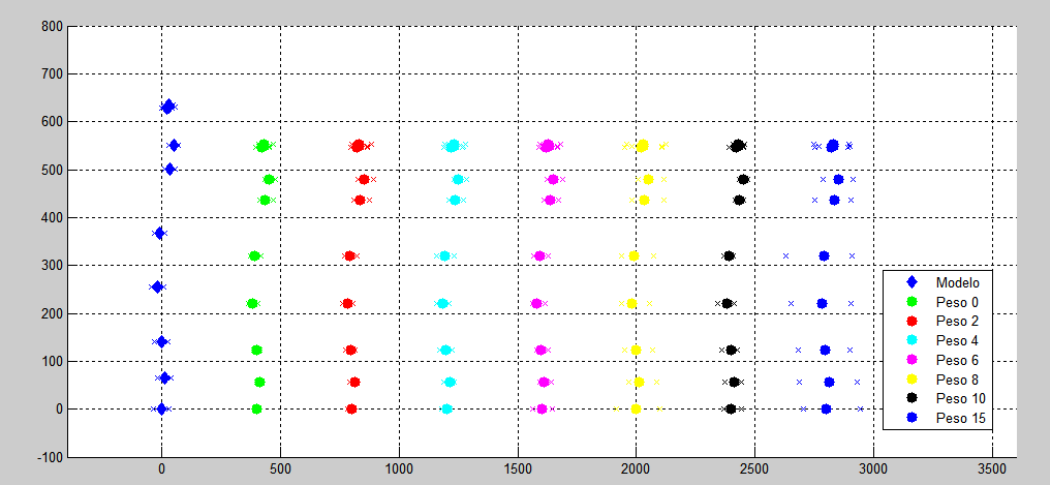


Ilustración 23. Ejemplos "Columna".

Ver código "Columna".

5.- Resultados.

Los resultados obtenidos al final de la ejecución de nuestro programa son dos documentos:

- El primero de ellos es el resultado de corregir, tratar y eliminar los artefactos de las señales del archivo cargado al principio, obteniendo una matriz de datos correctos y sin errores. Este primer documento es guardado en la carpeta "Datos Corregidos" con el mismo nombre del archivo original. Si ejecutamos el programa cargando el fichero "m5_00_13", al finalizar dicho programa, aparecerá en la carpeta "Datos Corregidos" un fichero "txt" con el nombre "m5_00_13". Dicho fichero se puede abrir y se pueden observar las columnas correspondientes a las tres coordenadas de los 26 puntos reflectantes y las filas, cuyo número puede depender en caso de haber cortado o no la señal. Es por ello que dicho archivo puede ser utilizado para visualizarlo en el programa "Ver_DPM" pero puede dar problemas si lo cargamos en "Program.fig" ya que éste último utiliza los ficheros originales que contienen todos ellos un número determinado de filas.

Punto 1			**Punto 2**			**……**	**Punto 26**			
X	Y	Z	X	Y	Z	……	X	Y	Z	**ΔT**
X_{1-1}	Y_{1-1}	Z_{1-1}	X_{2-1}	Y_{2-1}	Z_{2-1}	……	X_{26-1}	Y_{26-1}	Z_{26-1}	**ΔT**
X_{1-2}	Y_{1-2}	Z_{1-2}	X_{2-2}	Y_{2-2}	Z_{2-2}	……	X_{26-2}	Y_{26-2}	Z_{26-2}	**ΔT**
……	……	…..	……	……	…..	……	……	……	…..	……
……	…..	…..	……	…..	…..	……	……	……	……	……
……	…..	…..	……	…..	…..	……	……	……	……	……
X_{1-*}	Y_{1-*}	Z_{1-*}	X_{2-*}	Y_{2-*}	Z_{2-*}	……	X_{26-*}	Y_{26-*}	Z_{26-*}	ΔT

Tabla 3. Ejemplo del 1er Resultado obtenido. *El nº de filas es variable.

- El segundo será el DPM del primero, es decir, una vez tenemos el primer fichero con los datos corregidos, se ejecuta "Doble Paso Medio" para la obtención de un paso todavía más correcto y preciso. De esta manera lo que se busca es conseguir un único paso lo más conciso posible.

  Ha sido llevado a cabo utilizando el criterio de rodillas. El paso comienza cuando las rodillas se cruzan y termina cuando se vuelven a cruzar de la misma manera y en el mismo sentido. Si se toma como principio del paso cuando la pierna derecha adelanta y se cruza con la pierna izquierda, ha de finalizar el paso cuando esto vuelva a suceder y no justo en el siguiente cruce que será cuando la pierna izquierda

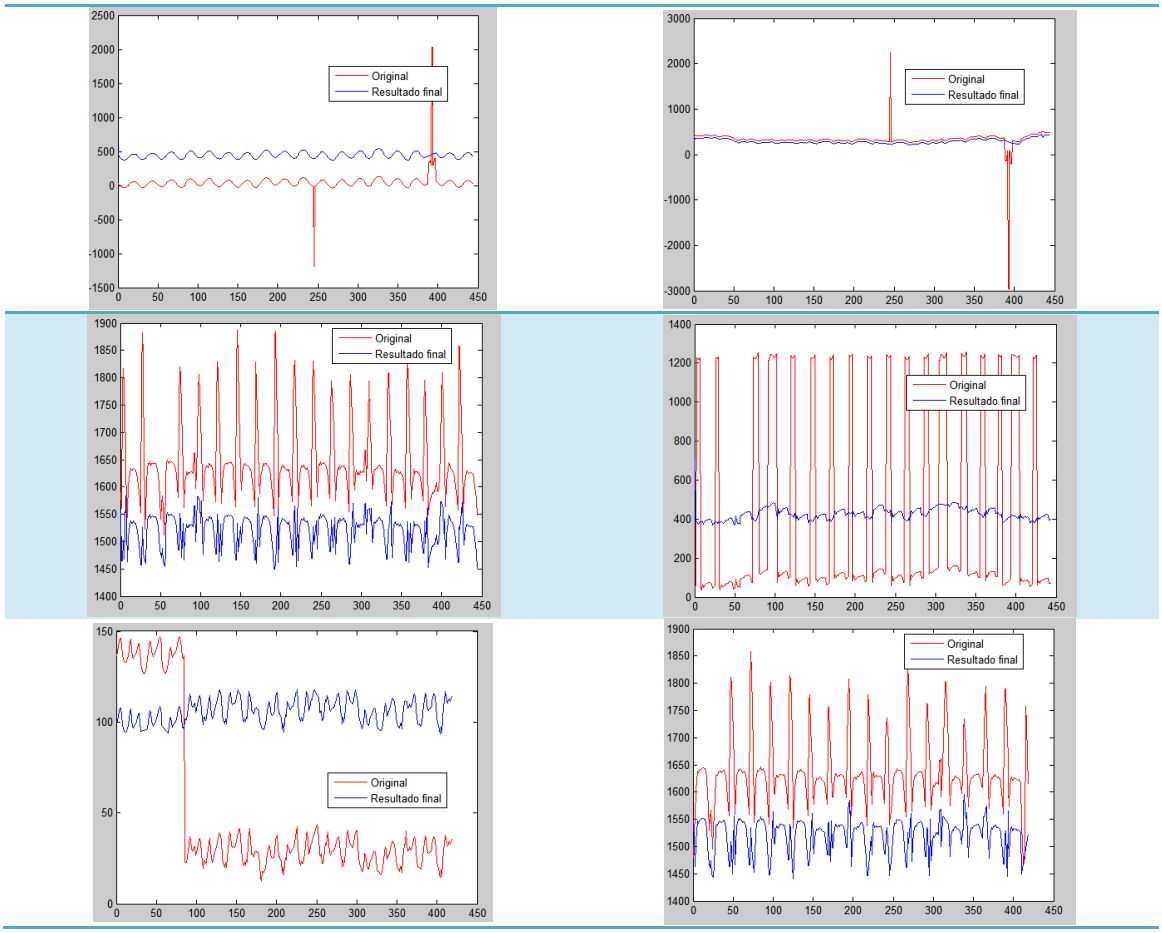
adelante a la derecha. De esta manera obtenemos el paso completo.

Como resultado obtenemos una matriz igual a la anterior pero con 100 filas, puesto que al ejecutar "DPM" obtenemos el paso en 100 muestras:

Punto 1			Punto 2			.....	Punto 26			
X	Y	Z	X	Y	Z	.....	X	Y	Z	ΔT
X_{1-1}	Y_{1-1}	Z_{1-1}	X_{2-1}	Y_{2-1}	Z_{2-1}	.....	X_{26-1}	Y_{26-1}	Z_{26-1}	ΔT
X_{1-2}	Y_{1-2}	Z_{1-2}	X_{2-2}	Y_{2-2}	Z_{2-2}	.....	X_{26-2}	Y_{26-2}	Z_{26-2}	ΔT
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
X_{1-100}	Y_{1-100}	Z_{1-100}	X_{2-100}	Y_{2-100}	Z_{2-100}	.....	X_{26-100}	Y_{26-100}	Z_{26-100}	ΔT

Tabla 4. Ejemplo del 2º Resultado obtenido.

En cuanto a los avances y logros conseguidos, a continuación veremos cómo se han limpiado y corregido las señales y cómo se ha pasado de no poder representar ni siquiera la figura de nuestro paciente a poder verlo e interpretarlo de una manera más sencilla.



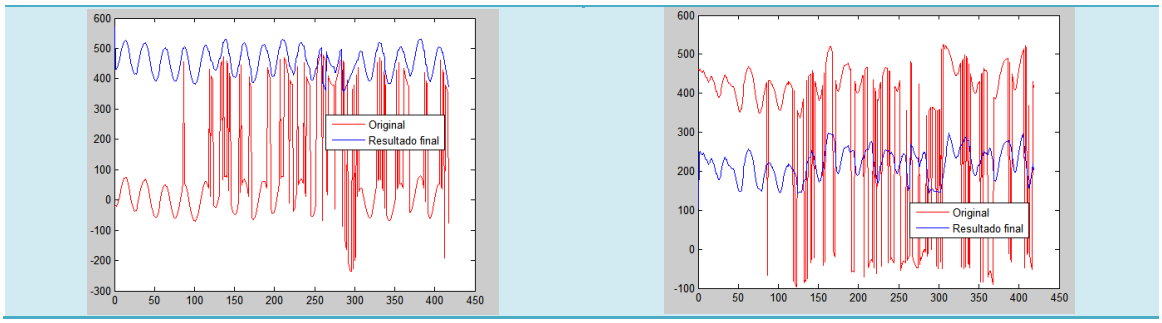


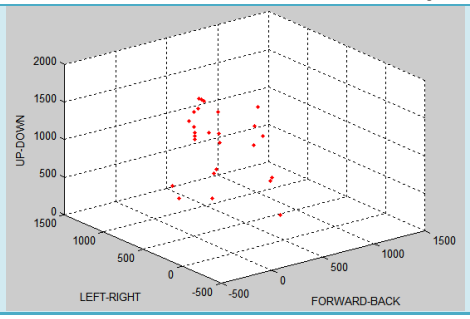
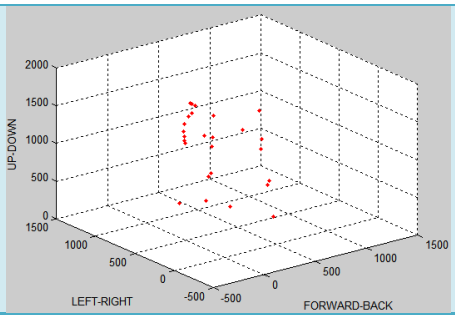
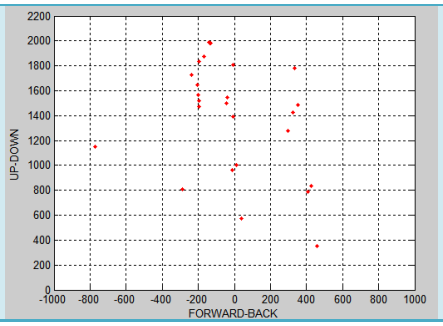
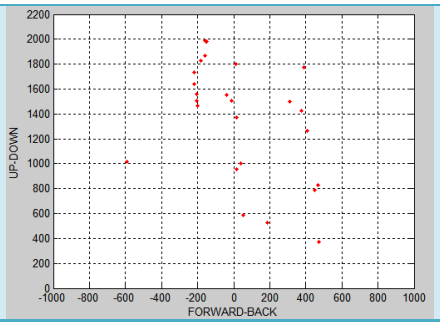
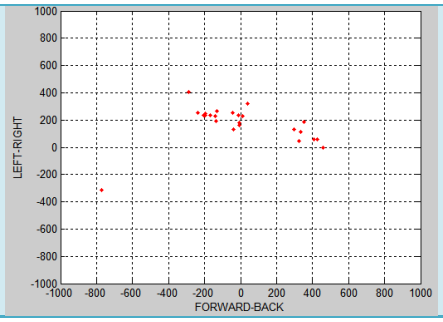
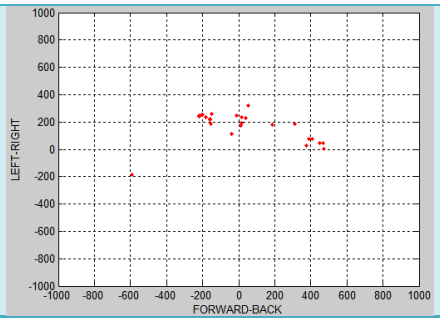
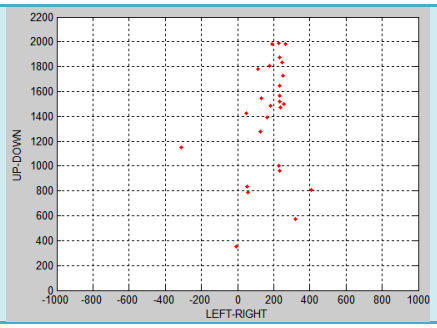
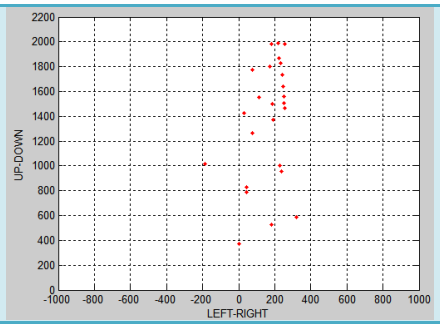
Tabla 5. Resultados. Ejemplos de señales corregidas.

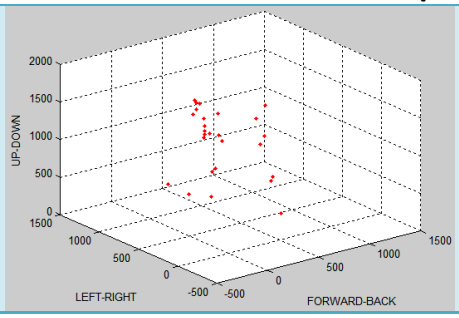
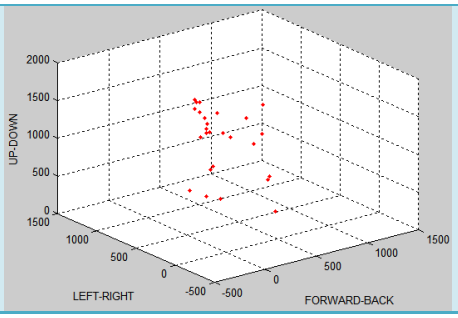
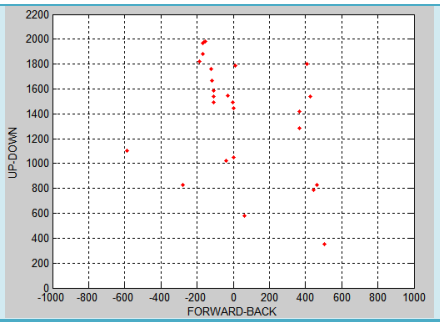
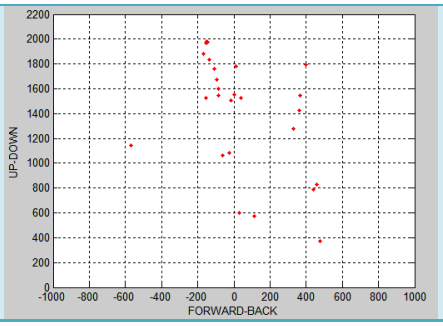
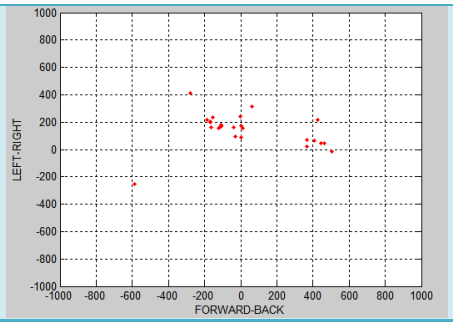
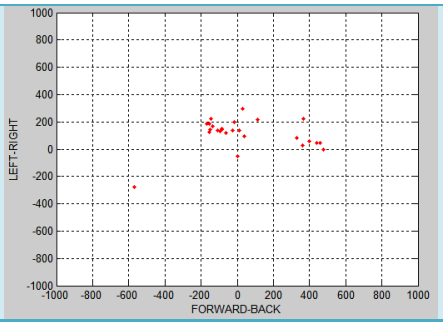
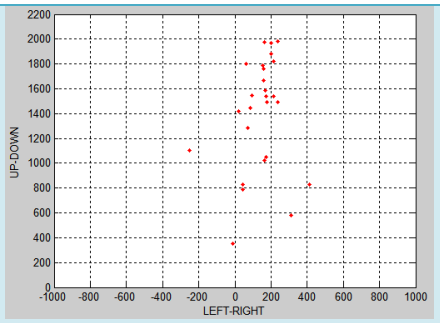
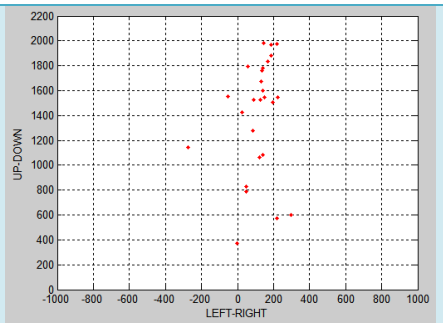
Centrándonos ahora en la representación 3D y los planos Y-Z, X-Z y Y-X se observa más fácilmente los problemas de los que hemos hablado anteriormente y el resultado que se ha conseguido.

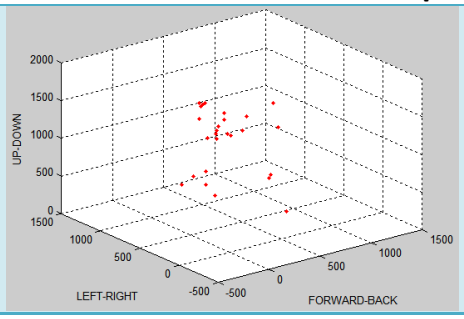
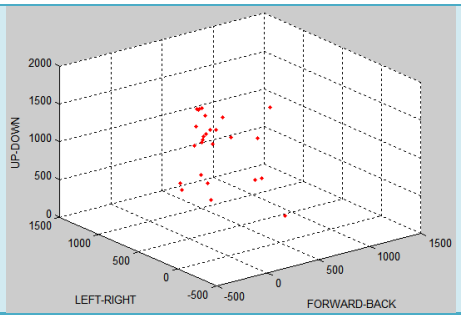
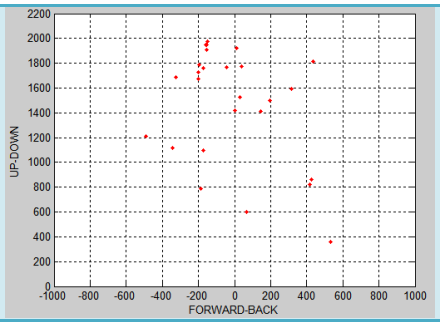
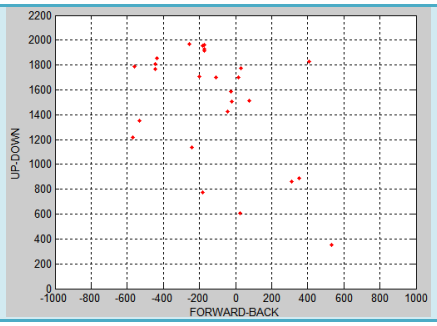
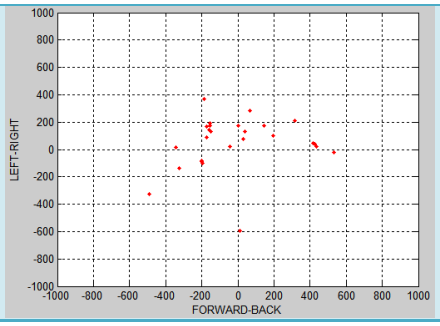
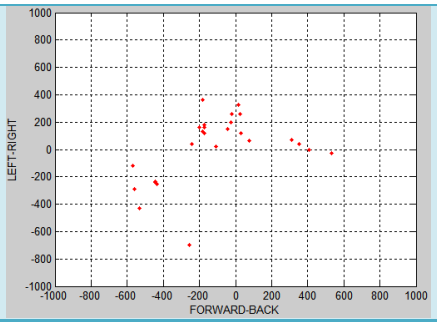
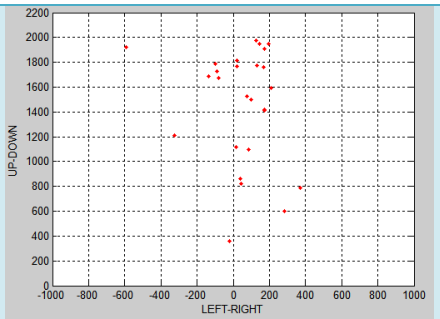
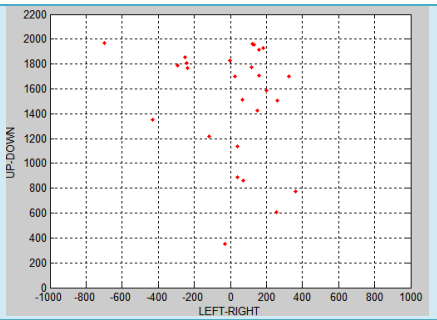
Se puede observar que a medida que se ha aumentado el peso, las señales recogidas en el laboratorio son peores y sus correcciones son más difíciles. El paciente no aparece centrado a la hora de representarlo, la columna no está a la altura de las piernas, el cuerpo sale torcido o de medio lado y en especial los pies sufren grandes reflexiones.

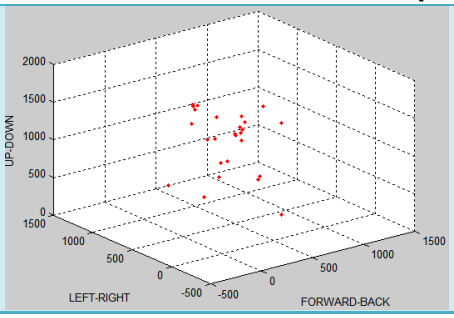
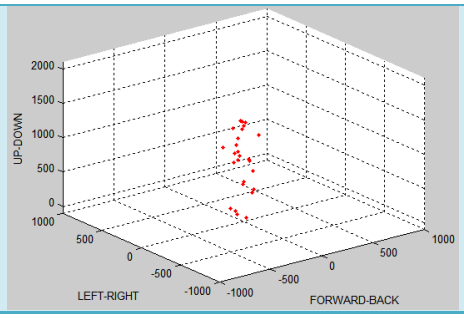
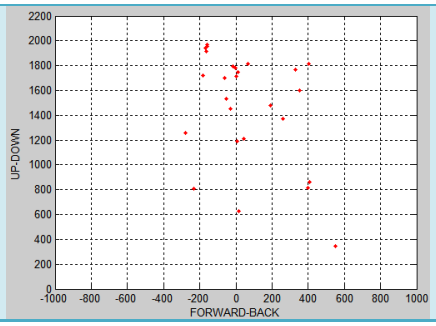
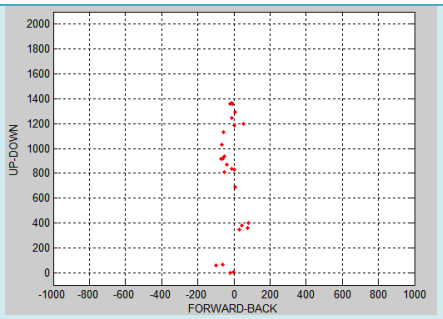
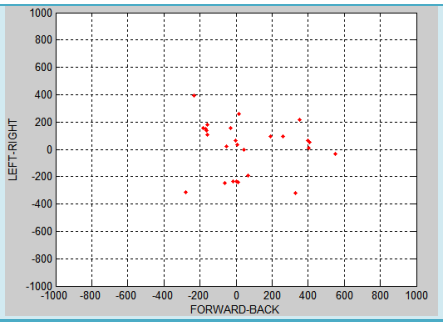
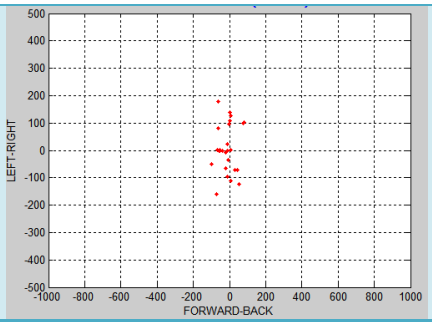
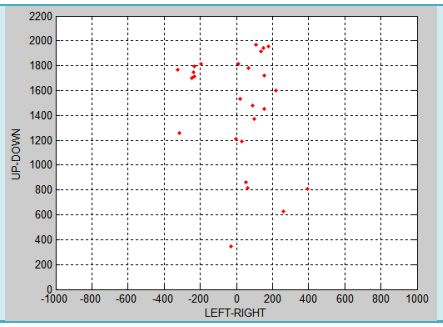
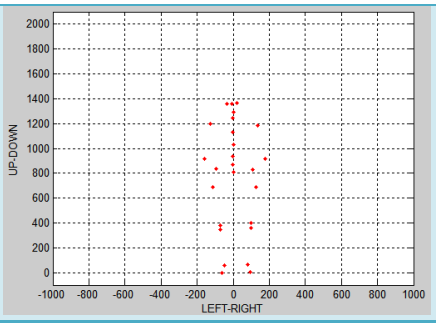
Los ficheros son del paciente Matthiew y se muestran a continuación desde el mínimo peso al máximo de manera ascendente en las siguientes tablas. El resultado está al final en la última tabla. En la práctica si cargamos los ficheros en el programa y lo vemos funcionando y en movimiento se puede ver y comprobar las correcciones más claramente pero aquí sobre el papel sólo podemos ver la figura sin movimiento y es por ello que sólo se muestra un resultado para todos los pesos, pues en realidad sólo estamos viendo la estática que es igual para todos los archivos, sólo variarían en este caso las proporciones.

Tabla 6. Comparación entre los datos originales y los resultados.

Peso 0	Peso 2
Ficheros originales	
Representación 3D	
	
Plano Y-Z	
	
Plano X-Z	
	
Plano Y-X	
	

Peso 4	Peso 6
Ficheros originales	
Representación 3D	
	
Plano Y-Z	
	
Plano X-Z	
	
Plano Y-X	
	

Peso 8	Peso 10
Ficheros originales	
Representación 3D	
	
Plano Y-Z	
	
Plano X-Z	
	
Plano Y-X	
	

Peso 15	Resultado
Fichero original	
Representación 3D	
	
Plano Y-Z	
	
Plano X-Z	
	
Plano Y-X	
	

6.- Resumen y conclusiones.

El proyecto consta básicamente de dos programas, uno para el tratamiento y corrección de las señales (“Program.fig”) y otro para la visualización de dichas señales (“Ver_DPM.fig”).

Los resultados obtenidos a través de los programas tienen como objetivo la interpretación que un médico puede tomar sobre el paciente sometido a la prueba, por lo que el médico ha de conocer el funcionamiento del segundo programa mencionado anteriormente mientras que no tiene por qué entender el primero de ellos. De esta forma separamos el tratamiento de señal de la interpretación médica.

Podemos concluir que el presente proyecto contribuye a un avance en la medicina puesto que con una simple prueba donde el paciente no padece ningún tipo de sufrimiento se es capaz de estudiar, analizar e interpretar la manera de andar del paciente y cómo dependiendo del peso que éste cargue puede afectar a la columna vertebral.

En cuanto a la estructura del código, un avance en la tecnología para la recogida de datos facilitaría en gran medida las correcciones de este programa, sobre todo en las señales que son “irrecuperables” pero dependiendo de los procesos seguidos podemos llegar a obtener muy buenos resultados. Por otra parte, todas las señales de las que disponemos son diferentes y los errores los hay de muchos tipos diferentes por lo que es casi imposible una perfecta corrección de los datos que represente perfectamente a nuestro paciente. En general, se observa que hay señales que pueden tratarse de distinto modo a otras, dificultando así la elaboración de un programa que corrija automáticamente todas las señales.

En definitiva, el presente proyecto favorece la representación de la realidad de manera rápida, cómoda y sobre todo útil, pudiendo así diagnosticar a un paciente sin que el mismo sea necesario durante el propio análisis de los resultados. Es un claro avance que junto con otros proyectos puede conseguir metas, ventajas y facilidades que hasta ahora no había.

7.- Agradecimientos.

En primer lugar me gustaría agradecer al profesor Jacek Dusza por haberme dado la oportunidad de trabajar en uno de sus proyectos y por su espléndido comportamiento hacia mí, haciéndome sentir muy cómodo y útil. A toda la gente que ha estado detrás de los proyectos anteriores a éste posibilitando la realización del mismo.

A Miguel Ángel Gómez Laso por haberme ayudado, guiado, recomendado y aconsejado en éste sexto año de carrera, todos mis profesores que me han instruido para que hoy esté aquí y en general a todo el mundo que ha hecho posible que este año haya disfrutado de un programa de movilidad internacional "Erasmus". Por ello agradezco que las dos universidades, tanto la Universidad Pública de Navarra (UPNA) como la Universidad Politechnika Warszawska hayan hecho esto posible, porque ha sido una experiencia que ha aportado cuantiosos valores a mi persona, no sólo académicamente hablando.

A todos mis amigos y compañeros de universidad, por todos esos buenos ratos que me han hecho pasar, haciendo más llevadero el trascurso de la carrera. A todos mis amigos más cercanos con los que más he compartido mi vida que siempre han estado ahí ayudándome en lo que han podido.

A toda mi familia que siempre ha estado para lo bueno y para lo malo y hago especialmente hincapié en mis padres, porque ellos han sido los que me han ayudado en todo momento, me han inculcado valores y por encima de todo es a ellos a quien les debo todo.

A toda esa gente y amigos que he conocido en éste año especial de Erasmus, con los que he compartido muchas horas y buenos momentos y a Alicia por haber sido un pilar fundamental.

En resumidas cuentas, agradecer a todo el mundo que de una manera u otra, directa o indirectamente me ha ayudado a ser lo que soy y estar donde estoy.

Han sido años de estudios difíciles pero a la vez muy gratificantes y satisfactorios, especialmente estos seis meses vividos en Varsovia.

Gracias a todos. Dziękuję bardzo.

8.- Bibliografía.

- http://www.mathworks.com/academia/student_center/tutorials/?s_cid=ML2012_tutorials

- http://www.mathworks.com/help/techdoc/creating_guis/bqz6p81.html

- http://www.mathworks.com/help/techdoc/creating_guis/f10-998674.html

- http://www.math.ufl.edu/help/matlab-tutorial/matlab-tutorial.html

- http://www.fimee.ugto.mx/profesores/rguzman/documentos/pie-guide.pdf

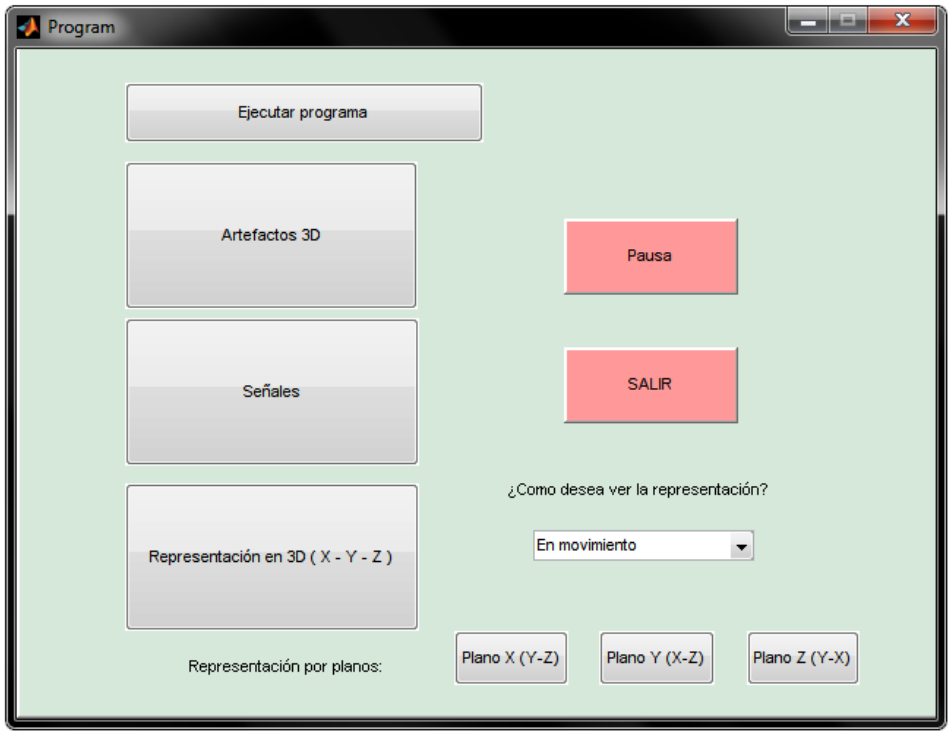
9.- Anexos.

1.- Manual de usuario.

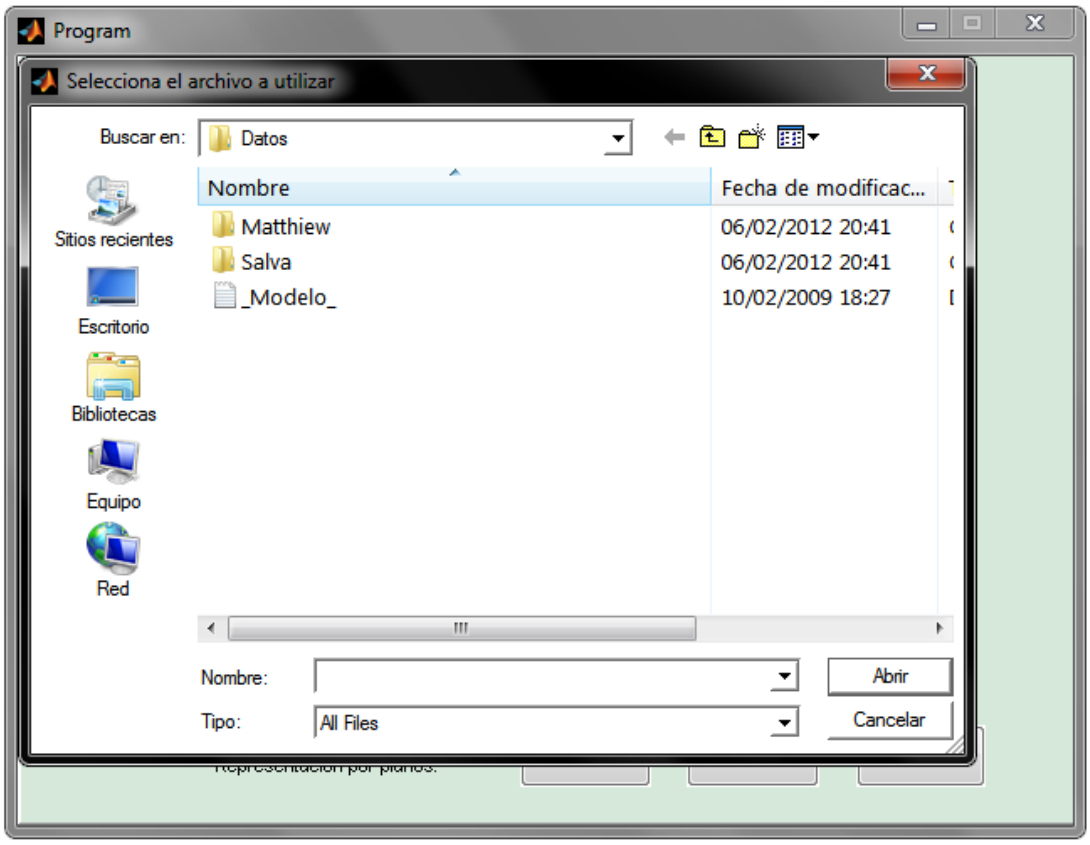
El programa está formado por varias subcarpetas y archivos donde tendremos dos opciones, tratar la señal para corregir las señales erróneas o simplemente visualizar cualquier archivo que deseemos, ya sea un resultado final proveniente de un archivo ya tratado o un archivo original. Si elegimos la primera opción hemos de arrancar el archivo "Program.fig" y en caso de la segunda opción el archivo "Ver_DPM.fig".

a) Program.fig

Al abrir el fichero "Program.fig" estaremos ejecutando la interfaz gráfica y aparecerá el siguiente cuadro de opciones:

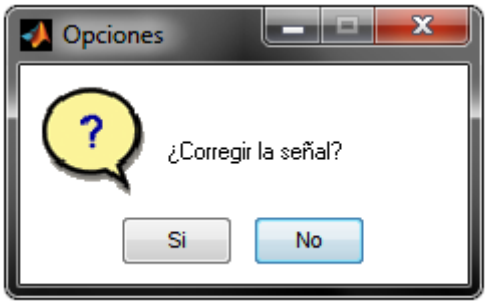


El primer paso será cargar el archivo que queramos tratar. Para ello pulsaremos en el botón “Ejecutar programa” y se nos abrirá una ventana para seleccionar el archivo “txt”.



A continuación el programa empezará a ejecutarse y nos saldrán varios cuadros emergentes:

Paso 1 - ¿Corregir la señal?



Si pulsamos en “No” el programa continuará al siguiente paso y si pulsamos en “Si” nos aparecerá el siguiente cuadro:

Nos muestra las posibles opciones que tenemos:

-"Pies del Modelo"

Como hemos visto en todos los experimentos y archivos de la recogida de datos, los principales problemas y donde más reflexiones hay son en los pies, por ello esta opción lo que hace es sustituir los pies de nuestro paciente por los de un modelo. El objetivo de ello es poder observar bien el conjunto de los 26 puntos. Sin embargo si lo que nos interesa del paciente es principalmente los pies, esta opción no deberíamos usarla.

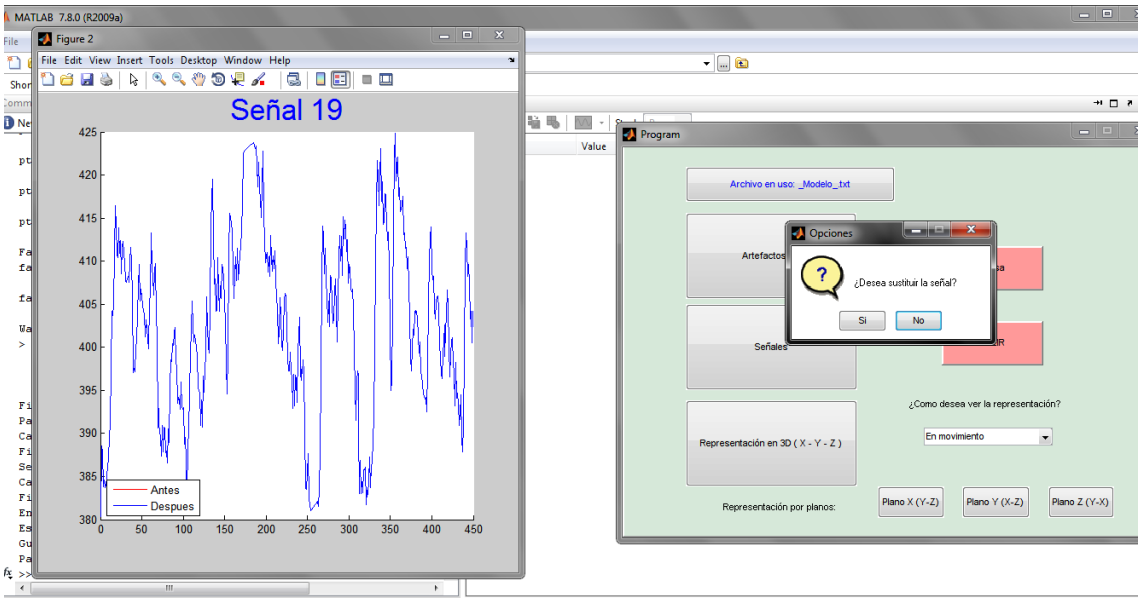
Una vez pulsado el botón el programa continuará con el siguiente paso.

-"Corregir todos los sensores"

Esta opción ejecuta un algoritmo para corregir más errores. Si elegimos esta opción el programa se ejecuta automáticamente y pasaremos al siguiente paso.

-"Corregir pies"

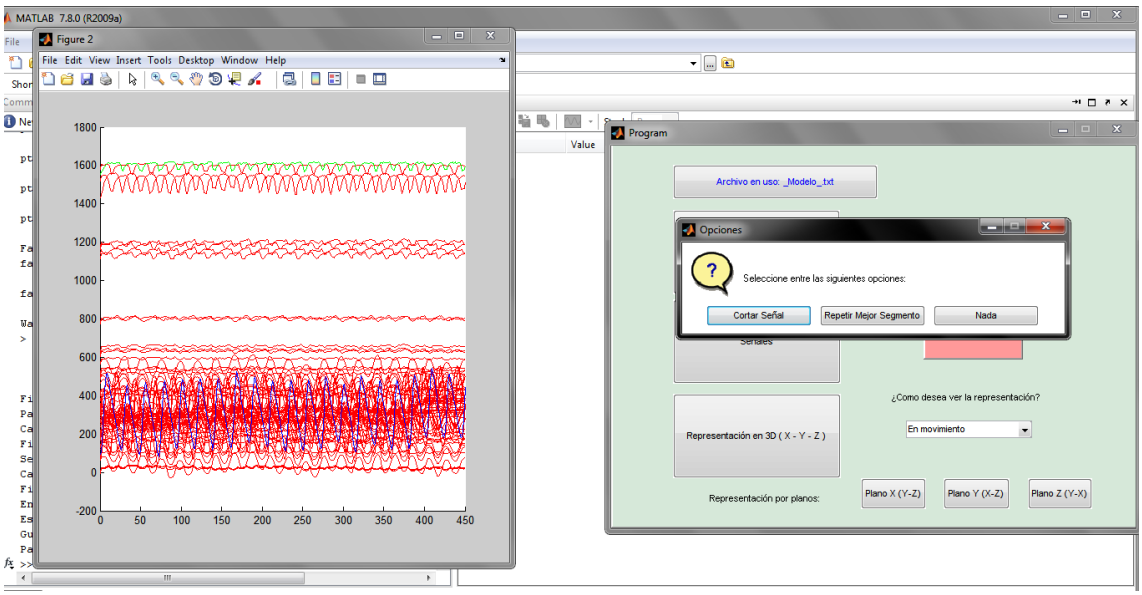
Como ya hemos comentado antes donde más problemas hemos tenido han sido en los pies, por ello esta opción permite corregir señal por señal (coordenada por coordenada) los puntos únicamente de los pies. Por lo tanto como tenemos 2 pies, 2 puntos por pie y 3 coordenadas por punto reflectante, tendremos un total de 12 posibles correcciones. Al pulsar dicho botón aparecerá la imagen donde se representa la señal antes y después de la corrección y un cuadro donde nos preguntará si deseamos sustituir o no la señal.



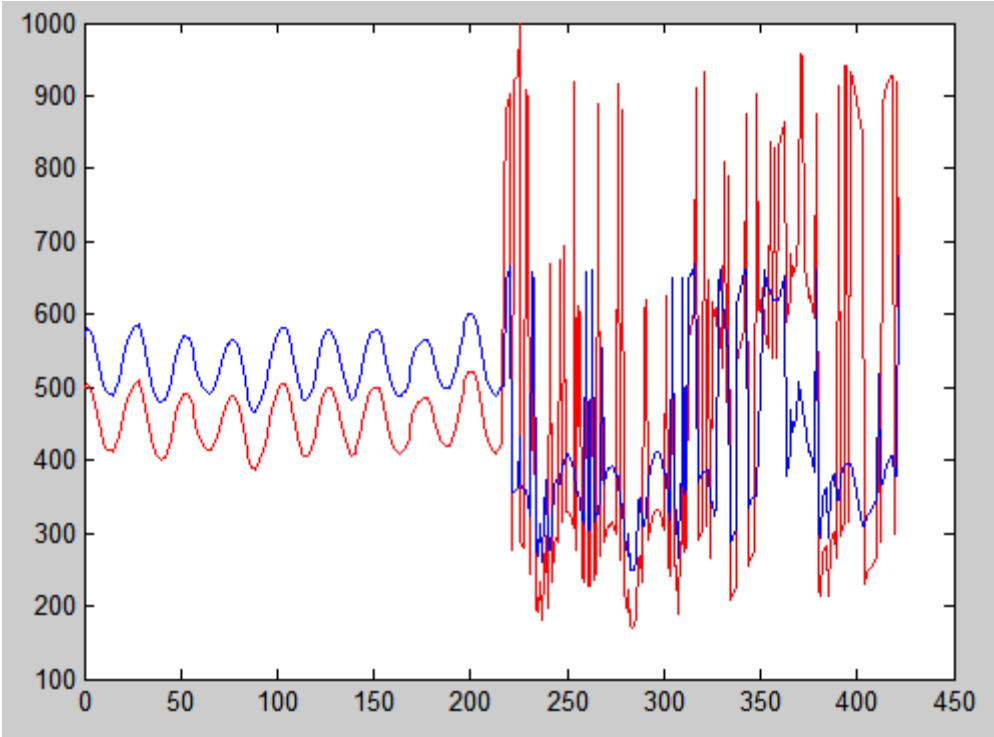
Después de esta primera opción donde podíamos corregir o no la señal nos saldrá la siguiente opción:

Paso 2 – Opciones

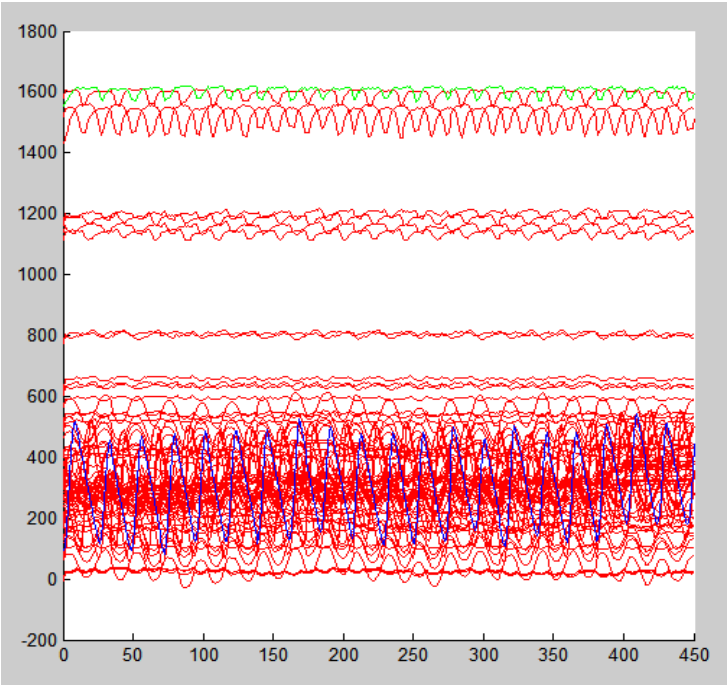
Este segundo paso constará de dos opciones principalmente, “Cortar Señal” y “Repetir Mejor Segmento”.

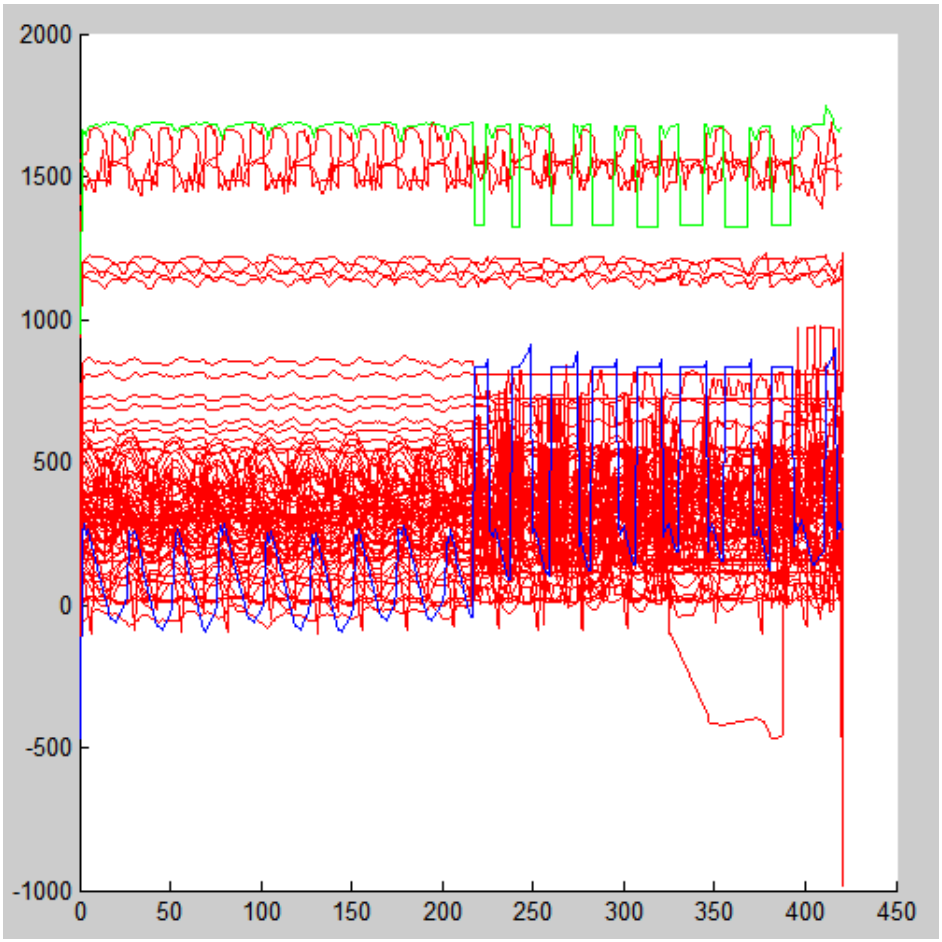


Dichas opciones se han incluido puesto que hay señales que debidas a una mala colocación del punto reflectante o a fallos en el sistema son prácticamente irrecuperables. Por ello en lo que consiste este paso es cortar la señal para quedarnos con la parte que nos interesa. La siguiente señal muestra este tipo de errores:

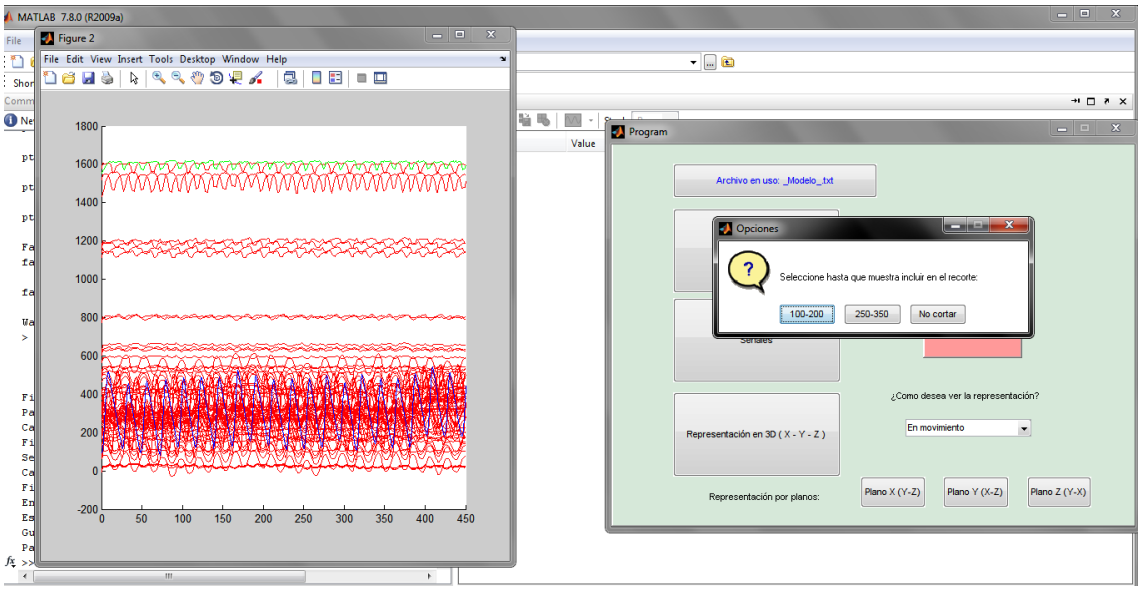


En la parte de la izquierda podemos ver que aparece una gráfica además. Dicha gráfica representa todas las señales superpuestas en el mismo gráfico para así poder ver si es necesario cortar o no la señal. A continuación podemos ver interpretaciones sobre una gráfica que representa señales buenas y otra donde las señales han sufrido errores al ser grabadas a partir de la muestra 200 aproximadamente.



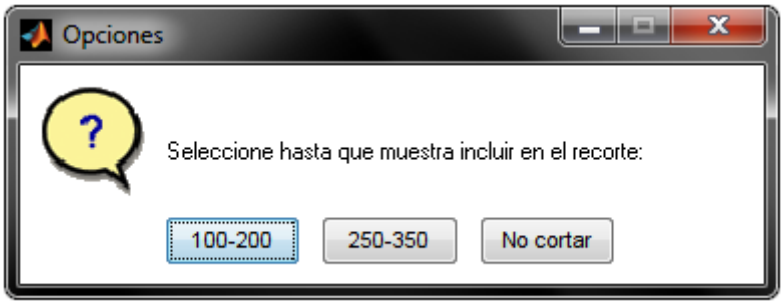


-“Cortar Señal”



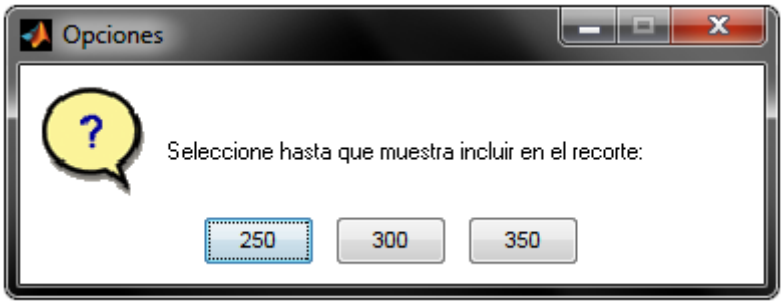
Esta opción nos permite comprobando la gráfica nombrada antes, cortar la señal por la muestra que indiquemos, pudiendo ser las opciones elegidas: 100, 150, 200, 250, 300 y 350.

El primer cuadro que nos saldrá al pulsar el botón “Cortar Señal” es el siguiente:



Si pulsamos en el botón “100-200” nos aparecerán las siguientes 3 opciones, 100, 150 y 200:

Si pulsamos sin embargo el botón “250-350”, tendremos las otras 3 opciones, 250, 300 y 350:



-“Repetir Mejor Segmento”

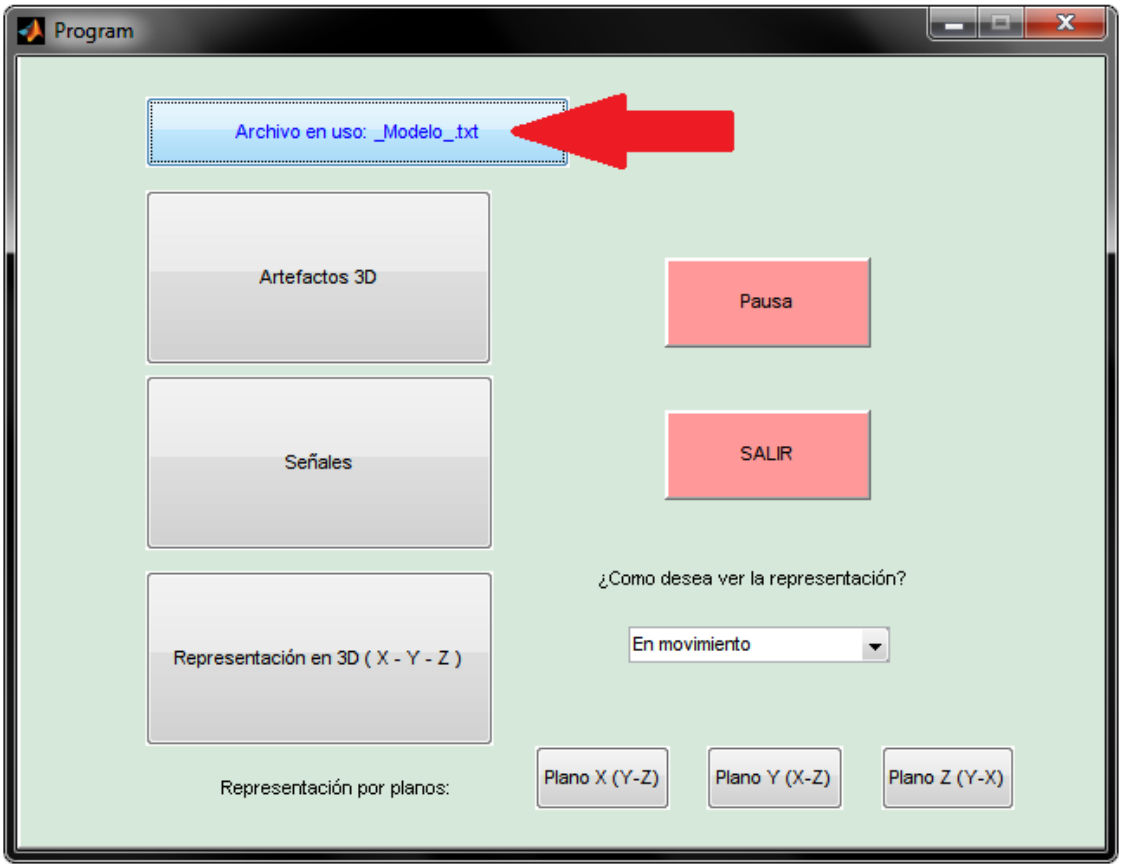
Este botón ejecuta un algoritmo el cual divide la señal en segmentos, determina la desviación típica de cada uno y se queda con el de menor. Con ello evitamos todos los picos, artefactos o reflexiones en nuestras señales. En

conclusión obtenemos el mejor segmento de la señal. Una vez que lo tenemos lo repetimos, lo concatenamos y así podemos minimizar los errores en los siguientes apartados.

-“Nada”

Esta opción es simplemente para cuando no queramos cortar la señal porque creemos que no es necesario o la señal no lo necesita.

Una vez ejecutado el programa, el archivo seleccionado anteriormente ocupará ahora el botón donde antes decía “Ejecutar programa”. Por ejemplo si hemos seleccionado “_Modelo_.txt” nos marcará “Archivo en uso: _Modelo.txt”. Si se desea cargar otro archivo bastaría pulsar de nuevo el mismo botón.

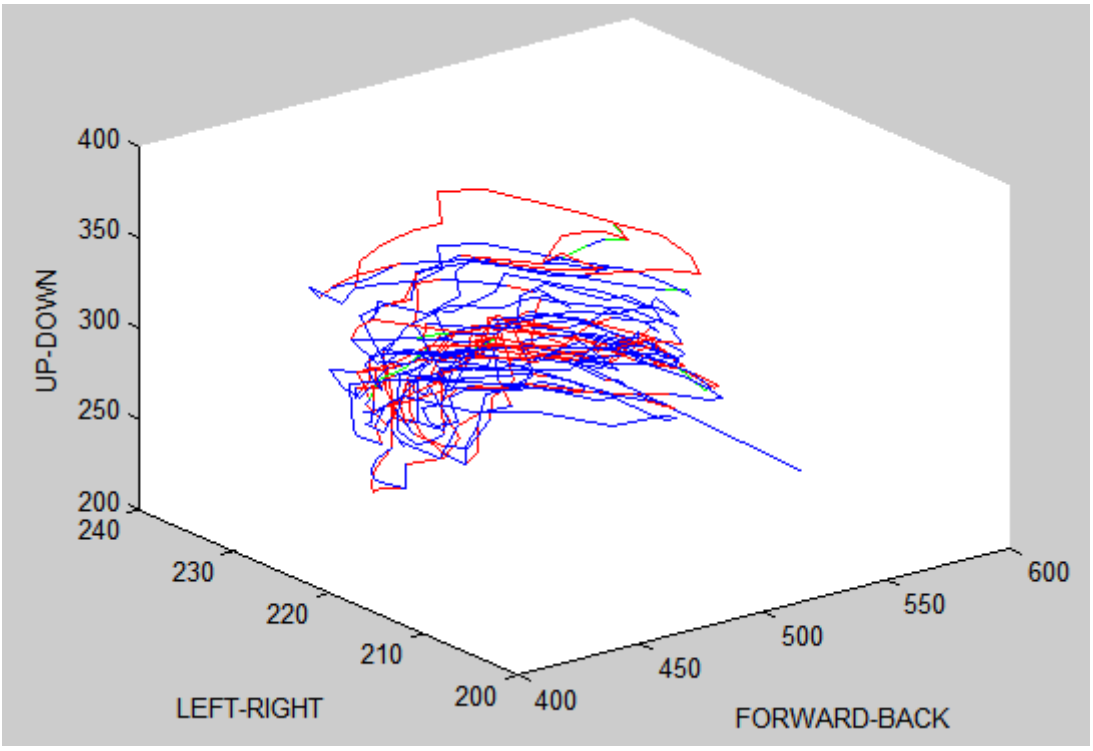


Una vez que ya hemos cargado nuestro archivo podremos utilizar los demás botones.

Nota: en todas las gráficas el color rojo representa la señal sin corregir y el azul la señal corregida. Puede verse en ocasiones una señal de color verde que refleja el paso intermedio entre la señal sin corregir y la corregida.

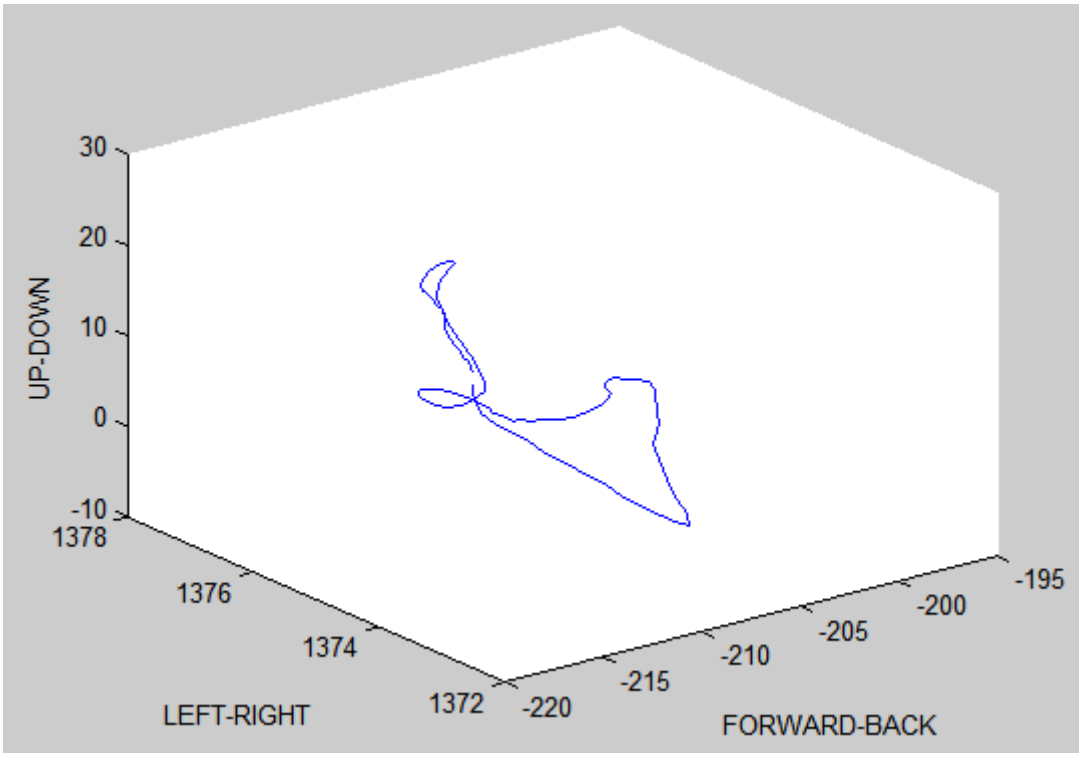
-Artefactos 3D

Muestra primero un cuadro donde nos da a elegir si queremos ver la señal completa o únicamente el "DPM" (Doble Paso Medio). A continuación se dibuja en un gráfico los datos del archivo cargado en 3 dimensiones y pulsando la tecla "Enter" podemos ir viendo punto por punto hasta completar los 26, donde el gráfico se cerrará automáticamente. El resultado pulsando en "Normal" es:



Tanto en este botón como en el siguiente, las señales en "Normal" que se visualizan son las que han seguido el proceso de tratamiento hasta la parte de "Paso_1_0_Corregir_artefactos", sin pasar por el resto de código puesto que así podemos ver cómo se corrigen automáticamente las señales completas.

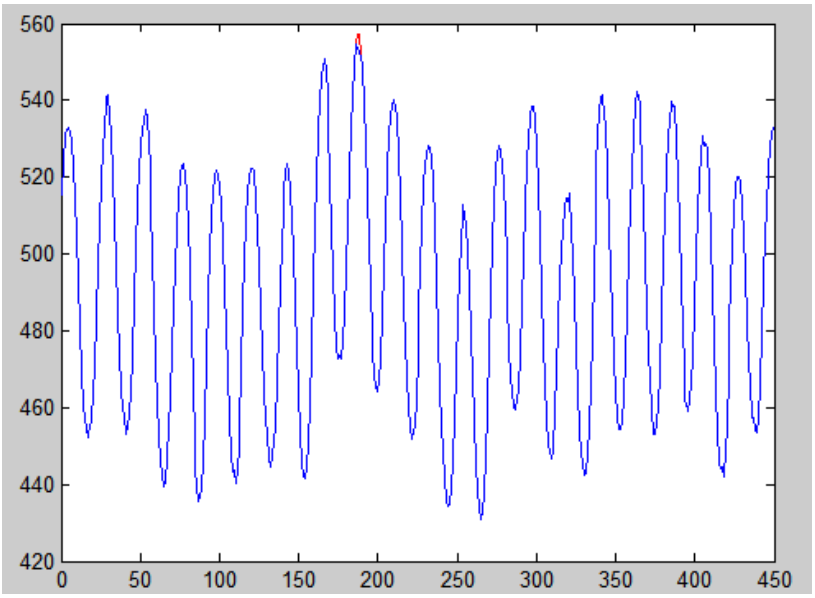
Y si pulsamos en “DPM”:

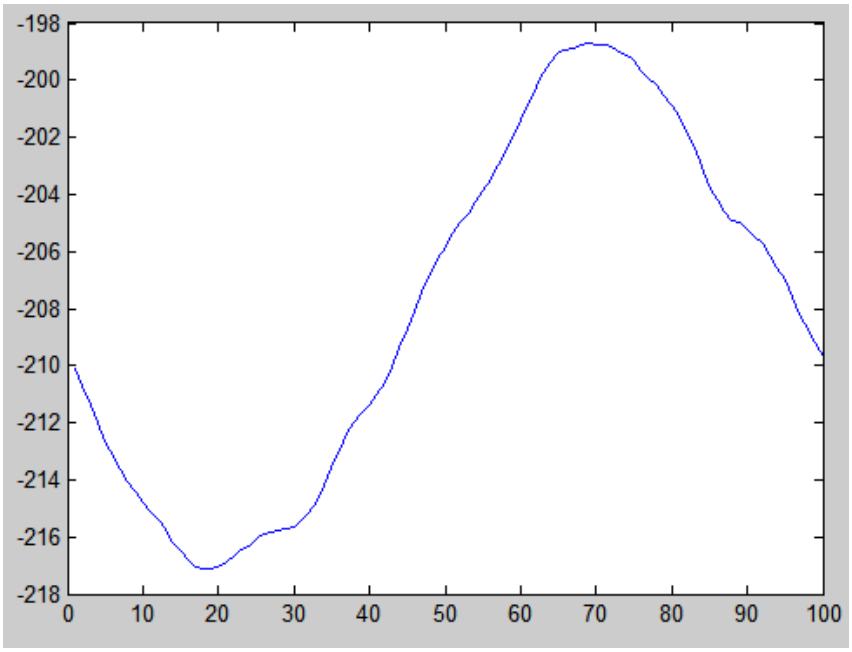


-Señales

Este botón es igual que el caso de Artefactos 3D con la diferencia de que aquí se muestran las señales en 2 dimensiones (tiempo o nº de muestra y espacio).

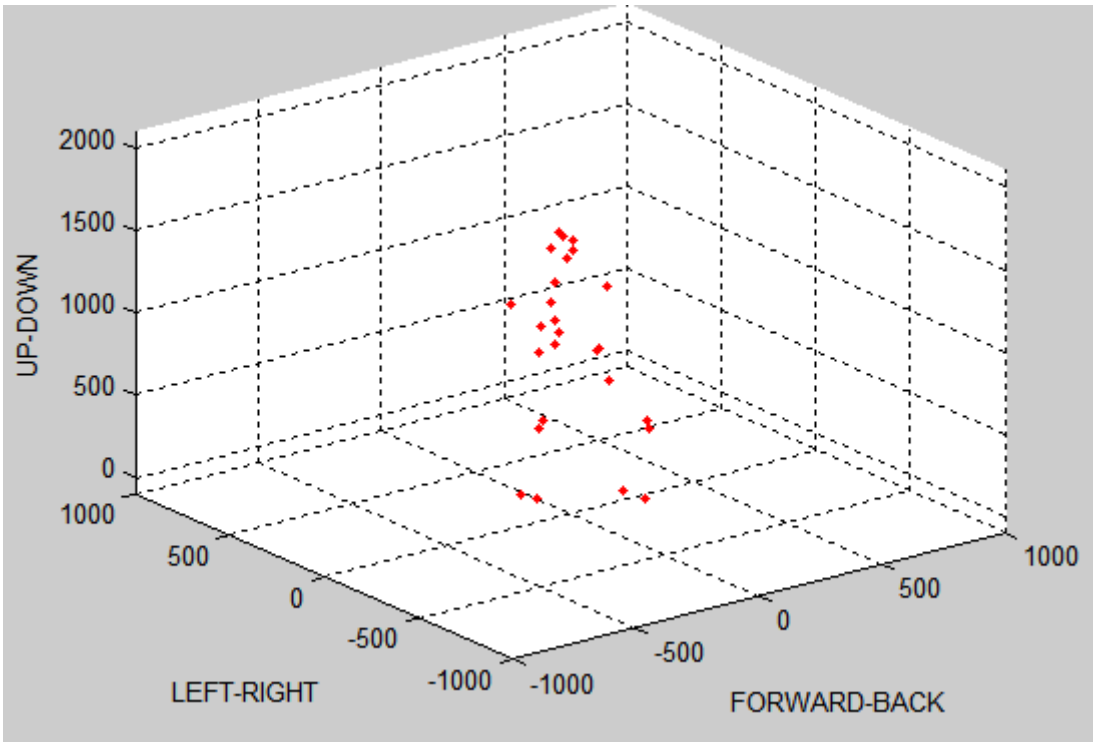
También aparece el cuadro emergente para elegir entre “Normal” o “DPM”:





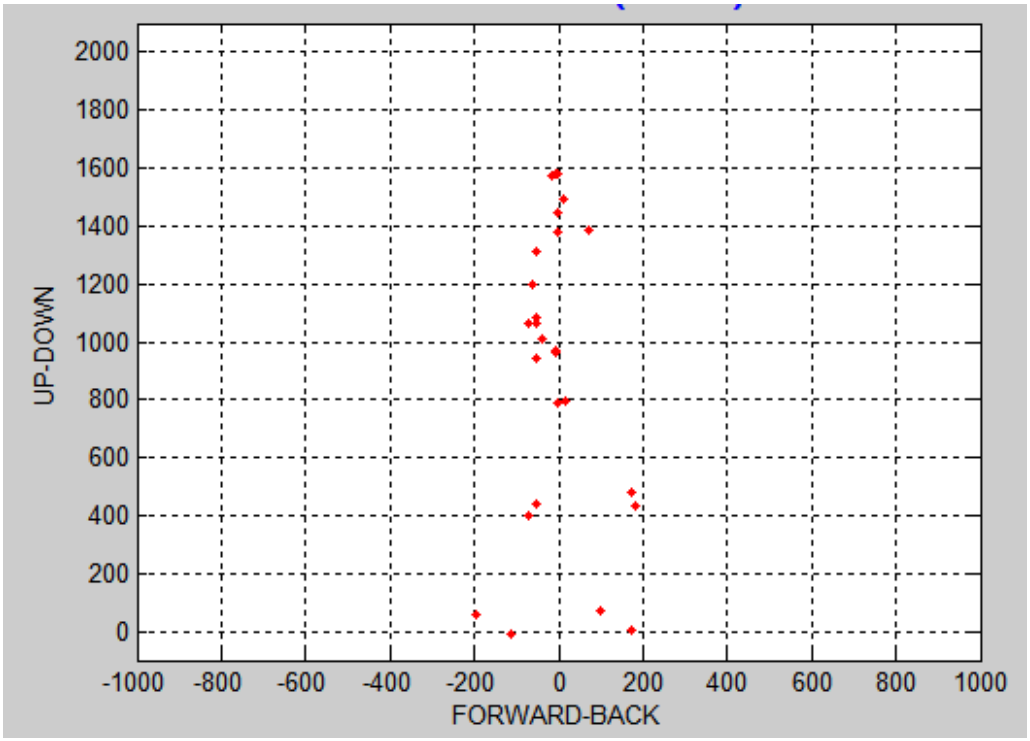
-Representación en 3D (X – Y – Z)

Mostrará los 26 puntos de nuestro paciente en 3 dimensiones y en movimiento.



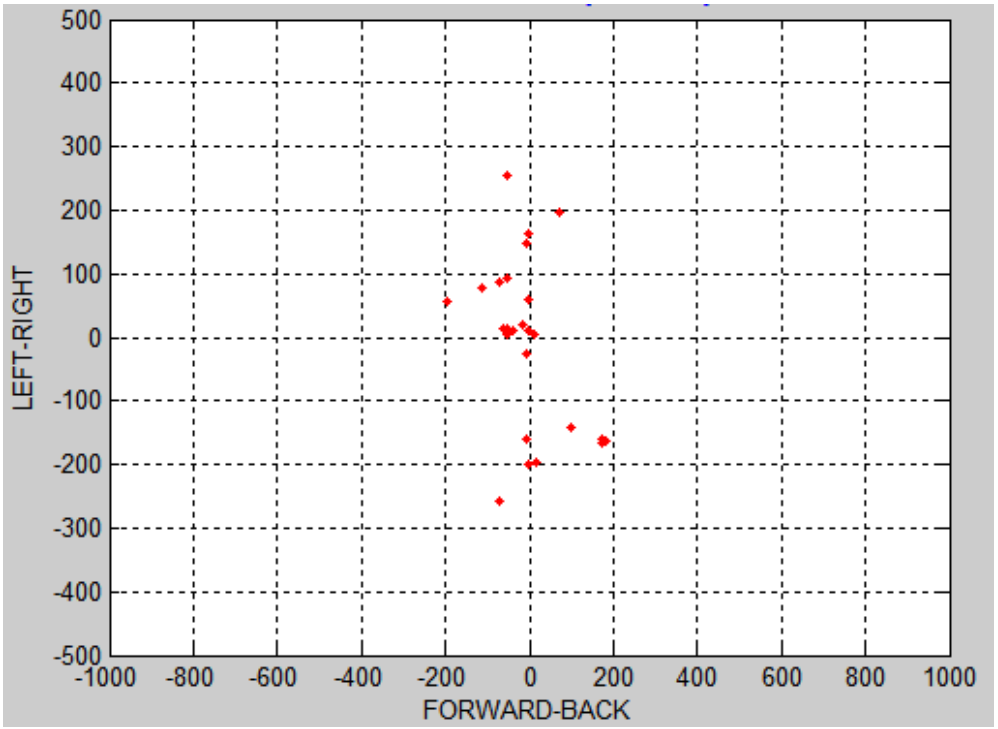
-Plano X (Y-Z)

Muestra el plano Y-Z, es decir veremos el lateral del paciente.



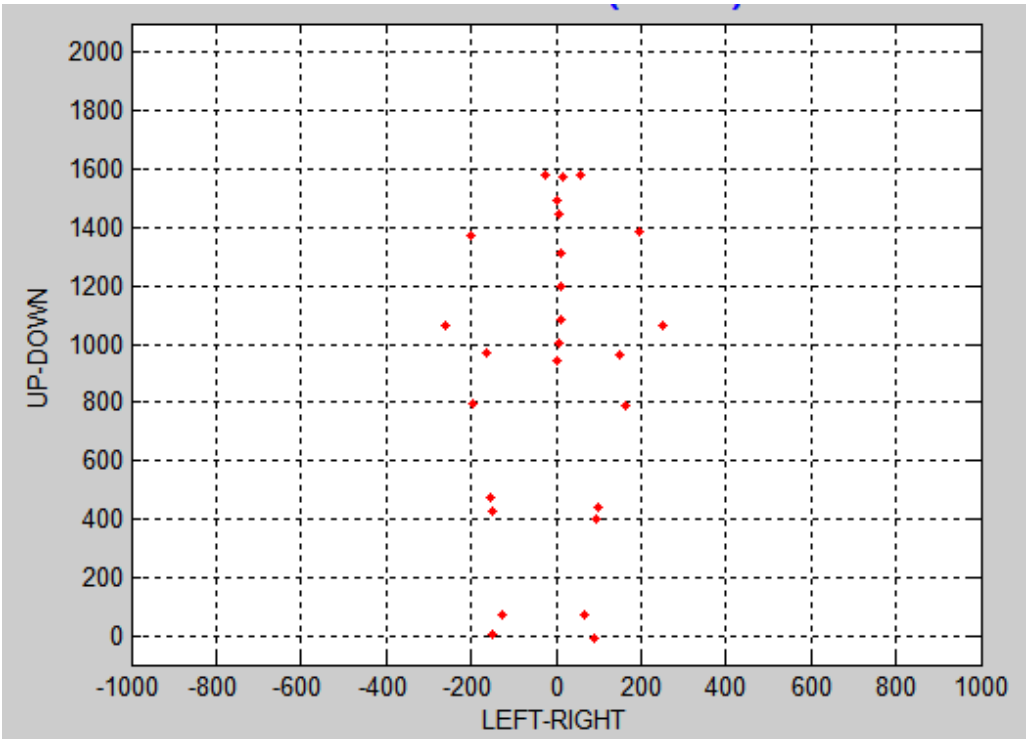
-Plano Y (X-Z)

Muestra el Plano X-Z, veremos a nuestro paciente desde arriba.



-Plano Z (Y-X)

Muestra el Plano Y-X, veremos a nuestro paciente de frente.



-¿Cómo desea ver la representación?

Esta opción permite que los botones de representación 3D y los planos podamos verlos en movimiento o de manera fija (estática).

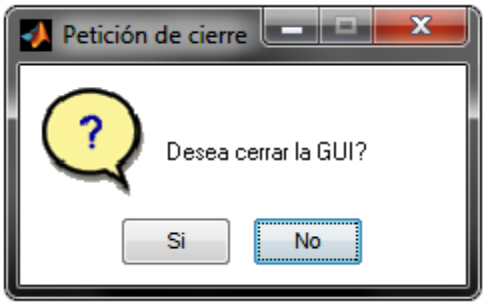
-Pausa

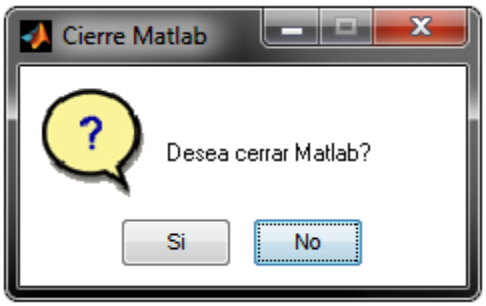
Este botón es fundamental para parar o salir de los bucles de los botones "Representación en 3D", "Plano X (Y-Z)", "Plano Y (X-Z)" y "Plano Z (Y-X)". Cuando pulsamos cualquiera de estos botones, aparecen las figuras en movimiento correspondientes y la forma adecuada de pararlas es pulsando el botón "Pausa".

-SALIR

Su propio nombre lo indica, dicho botón es utilizado para cerrar la interfaz gráfica y/o matlab también. Al pulsar el botón nos preguntará a través de cuadros emergentes lo que deseemos hacer. El primer cuadro nos pregunta sobre si queremos cerrar la "GUI" (Interfaz Gráfica). Si le decimos que si nos

muestra el cuadro de si deseamos cerrar matlab, pero si es que no volverá a la GUI.

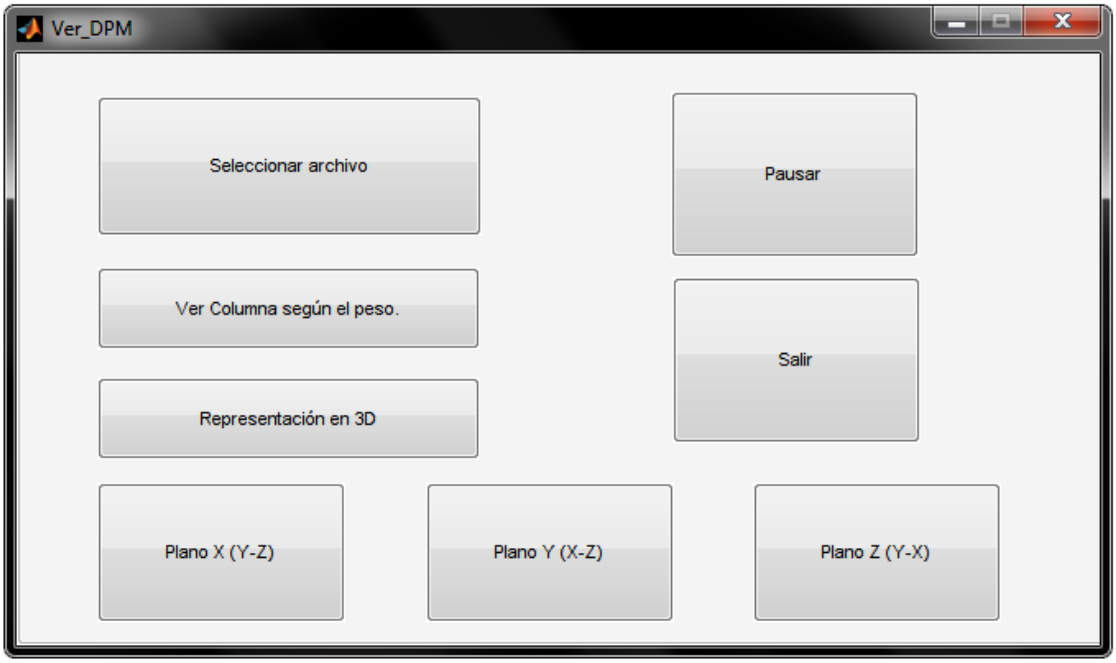




b) Ver_DPM.fig

Este programa consiste únicamente en la visualización, es por ello que es similar al anterior excluyendo las opciones de tratamiento de señal.

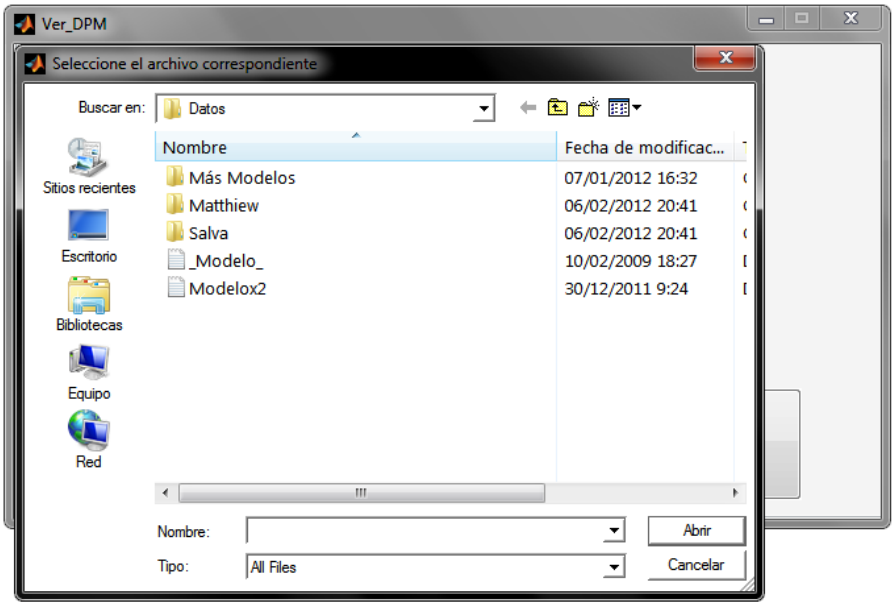
Por tanto al ejecutar este programa nos aparece un cuadro igual al de "Program.fig" pero sólo con las opciones de representación.



El funcionamiento de este programa es igual que el anterior. Primero ha de cargarse el archivo que queramos visualizar y una vez que nos diga que archivo está en uso, ya podemos representar o ver los planos que queramos. Las funciones de todos los botones son iguales que en el apartado anterior.

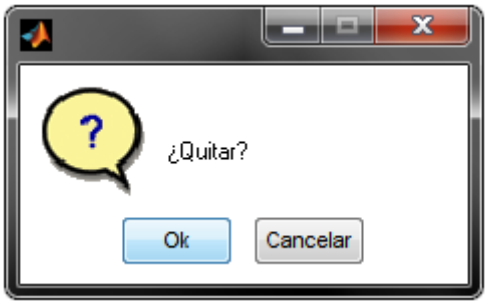
Sin embargo, podemos observar que hay un nuevo botón, "Ver Columna según el peso". Al pulsarlo, nos irá pidiendo que vayamos seleccionando los archivos correspondientes a los pesos, mostrando al final en una gráfica la columna de nuestro paciente con todos los pesos.

Como en el cuadro siguiente, nos irá pidiendo que vayamos seleccionando y abriendo los ficheros con todos los pesos.



Una vez repetido el paso anterior y habiendo seleccionado todos los ficheros con los pesos diferentes, nos aparecerá la gráfica con todas las representaciones y el siguiente cuadro, el cual hemos de pulsar para continuar a la siguiente representación.

Una vez pulsado el botón anterior, aparecerá la segunda gráfica mostrando las mismas representaciones pero lateralmente. Dichas representaciones vendrán acompañadas con el cuadro siguiente, para finalizar ya el proceso de representación de la columna de nuestro paciente.



2.- Advertencias y aclaraciones.

Para el correcto funcionamiento del programa es necesario, en todo momento que lo utilicemos, estar en la carpeta del programa, de lo contrario, si por un casual estamos en otra carpeta y pulsamos cualquier botón dará un error y el programa no funcionará. Para solucionarlo hemos de fijar la carpeta de matlab ("Current Directory") en la carpeta donde se encuentran los ficheros "Program.fig" y "Ver_DPM".

Los ficheros contienen código extra para facilitar la visualización de los pasos que se han realizado y por si fuera necesario utilizarlo por algún motivo.

El archivo "_Modelo_.txt" puede ser sustituido por otro. Para ello basta con sustituir el nombre del fichero escogido por _Modelo_.txt y colocarlo siempre en la carpeta de Datos, donde se encuentra ahora mismo.

En este proyecto el archivo utilizado como modelo ha sido "m033BMED.txt".

3.- Programación. Código.

A continuación mostramos el código principal. Dicho código es el encargado de generar la interfaz de usuario, las llamadas a los subprogramas comentados en el apartado 4 y los programas de representación en 3D, artefactos, señales y los planos según las coordenadas que se quieran visualizar.

0.- Código – Program (Programa Principal)

```
function varargout = Program(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Program_OpeningFcn, ...
                   'gui_OutputFcn',  @Program_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
movegui('east');                %La GUI se centra en la parte derecha de la
                                %pantalla
% End initialization code - DO NOT EDIT

function Ejemplo_OpeningFcn(hObject, eventdata, handles, varargin)
% Choose default command line output for Program
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

function varargout = Program_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%EJECUTAR PROGRAMA
function Ejecutar_Callback(hObject, eventdata, handles)

directorio_programa=pwd;
cd('data');

%Aplicamos corrección de artefactos + corrección enderezar cadera y cuerpo
Paso_1_0_Corregir_artefactos;

if nada==0

%Ejecutamos el Menú con las opciones de tratamiento de las señales, recortes...
Menu;

    %Para mostrar el archivo que estoy utilizando.
a=['Archivo en uso: ',archivo,char(10),'¿Cargar otro archivo?'];
```

```
set(hObject,'String',a);
set(hObject,'ForegroundColor','b');
guidata(hObject,handles)

%Guardamos la matriz buena antes de hacer el DPM.
%Para ello hemos de realizar los procesos de centrado
%y ensanchamiento.
datcorr_2=load('datcorr_2.txt');
datosbuenos=datcorr_2;
Paso_2_Centrar_cuerpo;
Paso_3_Factor_ensanchamiento_y_altura;
datcorr_2=load('datcorr_2.txt');
datcorr_2(:,79)=10;
cd(directorio_programa)
cd('Datos Corregidos')
texto=strcat(archivo);
save(texto, 'datcorr_2','-ascii')
cd ..
cd('data');
datcorr_2=datosbuenos;

%Aplicamos el programa DPM (Doble paso medio para suavizar el movimiento)
directorio1=pwd;
cd('DPM');
MainModificado;
datcorr_2=load('corregido.txt');
cd(directorio1);
save DPM.txt datcorr_2 -ascii
disp('Aplicado DPM');

%Ahora si hemos elegido la opción de sustituir por los pies del modelo,
%se ejecutarán las siguientes líneas para llevarlo a cabo.

p=load('piesmodelo');
if p==1
    disp('Sustituyendo los pies del modelo');
    DPM=load('DPM.txt');
    modelo=load('DPM_Pies_Modelo_.txt');
    modelo=modelo(1:size(DPM),:);
    DPM(:,19)=modelo(:,19);  %Ahora que tenemos los dos DPM, del modelo y
    DPM(:,20)=modelo(:,20);  %del paciente podemos sustituir los pies.
    DPM(:,21)=modelo(:,21);
    DPM(:,22)=modelo(:,22);
    DPM(:,23)=modelo(:,23);
    DPM(:,24)=modelo(:,24);
    DPM(:,73)=modelo(:,73);
    DPM(:,74)=modelo(:,74);
    DPM(:,75)=modelo(:,75);
    DPM(:,76)=modelo(:,76);
    DPM(:,77)=modelo(:,77);
    DPM(:,78)=modelo(:,78);
    save DPM.txt DPM -ascii
    save datcorr_2.txt DPM -ascii;
end

%Centramos el cuerpo entero (26 sensores en el centro de nuestras gráficas
%tomando como referencia los sensores de la columna vertebral.
%Corregimos la altura, coordenada Y.

Paso_2_Centrar_cuerpo;
```

```
%Como estamos tomando la estatica del modelo y no tienen porque
%medir ni de altura ni de anchura lo mismo las dos personas,
%aplicaremos dicho factor de correción.
%Restamos a los datos la estatica que teníamos para despues de este
%programa tenerlos con la estática buena

Paso_3_Factor_ensanchamiento_y_altura;

%Guardamos ahora los datos buenos

datcorr_2=load('datcorr_2.txt');
datcorr_2(:,79)=10;       %Ponemos la ultima columna en 10 (el tiempo).
save representacion.txt datcorr_2 -ascii; %Resultado final.
cd(directorio_programa);
mkdir('Datos Corregidos DPM');      %Guardamos el DPM
cd('Datos Corregidos DPM');
texto=strcat('DPM_',archivo);
save(texto, 'datcorr_2','-ascii')
end     %Es el end del if para que si no se selecciona archivo y se le
        %de a cancelar, el programa no haga nada.
cd(directorio_programa);
disp('Fin del programa');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ARTEFACTOS 3D
function artef3D_Callback(hObject, eventdata, handles)
directorio_programa=pwd;
cd('data');
artefacto3D = questdlg('Seleccione entre las siguientes
opciones:','Opciones','Normal','DPM','Normal');
switch artefacto3D
    case 'Normal',
        datorig=load('datorig.txt');
        datcorr_2=load('datcorr_antes.txt');
        datcorr=load('datcorr.txt');
        figure('Position',[82   185   560   420]); %Para obtener las coordenadas
de figure 1- fig1=figure  2- get(fig1,'Position')
        for i=1:26
            plot3(datorig(:,i),datorig(:,i+1),datorig(:,i+2),'r'); %(En ROJO los
ORIGINALES)
            hold on
            plot3(datcorr(:,i),datcorr(:,i+1),datcorr(:,i+2),'g'); %(En VERDE
los CORREGIDOS 1)
            hold on
            plot3(datcorr_2(:,i),datcorr_2(:,i+1),datcorr_2(:,i+2)); %(En AZUL
los CORREGIDOS 2)
            title(['Sensor ',int2str(i)],'fontsize',25,'Color','b');
            legend('Original','Paso intermedio','Resultado
final','Location','Best')
            Xlabel('FORWARD-BACK');
            Ylabel('LEFT-RIGHT');
            Zlabel('UP-DOWN');
            pause %Para que haga cada iteración del bucle cada vez que pulsemos
cualquier tecla.
            hold off
        end
        delete (2); %Elimina la ventana de la gráfica.
    case 'DPM'
        datcorr_2=load('datcorr_2.txt');
        figure('Position',[82   185   560   420]);
        for i=1:26
```

```
            plot3(datcorr_2(:,i),datcorr_2(:,i+1),datcorr_2(:,i+2));
            title(['Sensor ',int2str(i)],'fontsize',25,'Color','b');
            Xlabel('FORWARD-BACK');
            Ylabel('LEFT-RIGHT');
            Zlabel('UP-DOWN');
            pause
        end
        delete (2);
end
cd(directorio_programa);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ARTEFACTOS 2D
function artef2D_Callback(hObject, eventdata, handles)
directorio_programa=pwd;
cd('data');
artefacto2D = questdlg('Seleccione entre las siguientes
opciones:','Opciones','Normal','DPM','Normal');
switch artefacto2D
    case 'Normal',
        datorig=load('datorig.txt');
        datcorr_2=load('datcorr_antes.txt');
        datcorr=load('datcorr.txt');
        figure('Position',[82   185   560   420]);
        for i=1:3:78

            plot(datorig(:,i),'r');  %Coordenada X del Sensor i
            hold on                  %Los datos originales y los corregidos.
            plot(datcorr(:,i),'g');
            hold on
            plot (datcorr_2(:,i));
            title(['Coordenada X Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            legend('Original','Paso intermedio','Resultado
final','Location','Best')
            pause
            hold off

            plot(datorig(:,i+1),'r'); %Coordenada Y del Sensor i
            hold on                   %Los datos originales y los corregidos.
            plot(datcorr(:,i+1),'g');
            hold on
            plot (datcorr_2(:,i+1));
            title(['Coordenada Y Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            legend('Original','Paso intermedio','Resultado
final','Location','Best')
            pause
            hold off

            plot(datorig(:,i+2),'r'); %Coordenada Z del Sensor i
            hold on                   %Los datos originales y los corregidos.
            plot(datcorr(:,i+2),'g');
            hold on
            plot (datcorr_2(:,i+2));
            title(['Coordenada Z Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            legend('Original','Paso intermedio','Resultado
final','Location','Best')
            pause
            hold off
```

```
        end
        delete (2);
    case 'DPM'
        datcorr_2=load('datcorr_2.txt');
        figure('Position',[82   185   560   420]);
        for i=1:3:78

            plot (datcorr_2(:,i)); %Coordenada X del Sensor i
            title(['Coordenada X Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause

            plot (datcorr_2(:,i+1));  %Coordenada Y del Sensor i
            title(['Coordenada Y Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause

            plot (datcorr_2(:,i+2));  %Coordenada Z del Sensor i
            title(['Coordenada Z Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause
        end
        delete (2);
end
cd(directorio_programa);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%REPRESENTACIÓN EN 3D
function repre3D_Callback(hObject, eventdata, handles)
directorio_programa=pwd;
cd('data');
a=load('representacion.txt');
figure('Position',[82   185   560   420]);
    n=size(a,1);
pausabucle=0;
save pausarepresentacion pausabucle -ascii
cd(directorio_programa);
x=n;
q=1;
                        while q<x
                            cd('data');
                            pausabucle=load('pausarepresentacion');
                            cd(directorio_programa);
                            if pausabucle==0
                                j=1:26;
                                r=j-1;
                                plot3(a(q,3*r+3),a(q,3*r+1),(a(q,3*r+2)),'r.');
                                axis([-1000 1000 -1000 1000 -100 2100]);
                                grid on
                                title('3D','fontsize',25,'Color','b');
                                Xlabel('FORWARD-BACK');
                                Ylabel('LEFT-RIGHT');
                                Zlabel('UP-DOWN');
                                drawnow
                            else
                                delete(2)
                                break
                            end
                            a(n+q,:)=a(q,:); %Ponemos este comando para que no
se acabe de reproducir hasta pulsar pausa.
                            x=x+1;
```

```
                            q=q+1;
                            pause((a(q,79)/1000));
                        end
cd(directorio_programa);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%EN MOVIMIENTO O DE MANERA FIJA

function movimiento_fijo_Callback(hObject, eventdata, handles)
directorio_programa=pwd;
cd('data');
val = get(hObject,'Value');
switch val
    case 1  %En movimiento
        datcorr_2=load('datcorr_2.txt');
        datcorr_2(:,79)=10;
        save representacion.txt datcorr_2 -ascii;
    case 2  %De manera fija
        estatica=load('estatica.txt');
        save representacion.txt estatica -ascii;
end
cd(directorio_programa);

function movimiento_fijo_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%COORDENADA PlanoX
function PlanoX_Callback(hObject, eventdata, handles)
directorio_programa=pwd;
cd('data');
a=load('representacion.txt');
n=size(a,1);
figure('Position',[82   185   560   420]);
pausabucle=0;
save pausarepresentacion pausabucle -ascii
cd(directorio_programa);
x=n;
q=1;
                        while q<x
                            cd('data');
                            pausabucle=load('pausarepresentacion');
                            cd(directorio_programa);
                            if pausabucle==0
                                        j=1:26;
                                        r=j-1;
                                        plot(a(q,3*r+3),(a(q,3*r+2)),'r.');
                                        axis([-1000 1000 -100 2100]);
                                        grid on
                                        title('Plano X (Y-
Z)','fontsize',25,'Color','b');
                                        Xlabel('FORWARD-BACK');
                                        Ylabel('UP-DOWN');
                                        drawnow
                            else
                                delete(2)
                                break
                            end
```

```
                         a(n+q,:)=a(q,:);
                         x=x+1;
                         q=q+1;
                         pause((a(q,79)/1000));
                     end
cd(directorio_programa);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%COORDENADA PlanoY

function PlanoY_Callback(hObject, eventdata, handles)
directorio_programa=pwd;
cd('data');
a=load('representacion.txt');
n=size(a,1);
figure('Position',[82   185   560   420]);
pausabucle=0;
save pausarepresentacion pausabucle -ascii
cd(directorio_programa);
x=n;
q=1;
                     while q<x
                         cd('data');
                         pausabucle=load('pausarepresentacion');
                         cd(directorio_programa);
                         if pausabucle==0
                                 j=1:26;
                                 r=j-1;
                                 plot(a(q,3*r+3),a(q,3*r+1),'r.');
                                 axis([-1000 1000 -500 500]);
                                 grid on
                                 title('Plano Y (X-
Z)','fontsize',25,'Color','b');
                                 Xlabel('FORWARD-BACK');
                                 Ylabel('LEFT-RIGHT');
                         else
                             delete(2)
                             break
                         end
                         a(n+q,:)=a(q,:);
                         x=x+1;
                         q=q+1;
                         pause((a(q,79)/1000));
                     end
cd(directorio_programa);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%COORDENADA PlanoZ

function PlanoZ_Callback(hObject, eventdata, handles)
directorio_programa=pwd;
cd('data');
a=load('representacion.txt');
n=size(a,1);
figure('Position',[82   185   560   420]);
pausabucle=0;
save pausarepresentacion pausabucle -ascii
cd(directorio_programa);
x=n;
q=1;
                     while q<x
```

```
                            cd('data');
                            pausabucle=load('pausarepresentacion');
                            cd(directorio_programa);
                            if pausabucle==0
                                    j=1:26;
                                    r=j-1;
                                    plot(a(q,3*r+1),(a(q,3*r+2)),'r.');
                                    axis([-1000 1000 -100 2100]);
                                    grid on
                                    title('Plano Z (Y-
X)','fontsize',25,'Color','b');
                                    Xlabel('LEFT-RIGHT');
                                    Ylabel('UP-DOWN');
                                    drawnow
                            else
                                delete(2)
                                break
                            end
                            a(n+q,:)=a(q,:);
                            x=x+1;
                            q=q+1;
                            pause((a(q,79)/1000));
                        end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%SALIR

function Salir_Callback(hObject, eventdata, handles)

selection = questdlg('Desea cerrar la GUI?','Petición de
cierre','Si','No','No');
switch selection
    case 'Si',
        delete(gcf)
    case 'No'
        return
end
selection2 = questdlg('Desea cerrar Matlab?','Cierre Matlab','Si','No','No');
switch selection2
    case 'Si',
        quit
    case 'No'
        return
end

function pausabucle_Callback(hObject, eventdata, handles)
directorio_programa=pwd;
cd('data');
pausabucle=1;
save pausarepresentacion pausabucle -ascii
cd(directorio_programa);

function fondo_Callback(hObject, eventdata, handles)
%Esto esta solo para evitar el problema del fondo al pausar las
%representaciones ya que sino se nos quedaba de fondo la última imagen.
```

Volver

1.- Código - Paso_1_0_Corregir_artefactos.

```
%Corremigos todos los artefactos.

directorio1=pwd;        %Guardamos el directorio original.
cd(directorio_programa);
cd('Datos');   %Nos situamos en el directorio de los datos originales.
[archivo,directorio2] = uigetfile('*.*','Selecciona el archivo a utilizar');
if archivo == 0  %Para que si le damos a cancelar no falle matlab.
    cd(directorio1);
    nada=1;
    return
else
    cd(directorio2);  %Vamos al directorio donde esta el archivo.
    a=load(archivo);  %Cargamos el fichero.
    nada=0;
    cd(directorio1);  %Volvemos al directorio donde tenemos los programas.
end

disp('Paso_1_0_Corregir_artefactos');

media=mean(a);          %Hallamos la media de las columnas.
media2=media;
m=size(a,1)-1;

for i=1:m   %Elaboramos una matriz con las medias para restarla luego.
    media=[media;media2];
end

datorig=a;  %Para compararlos más tarde en las gráficas.

%PRIMER PASO - Corregimos las señales con grandes reflexiones.
%(principalmente los pies)

senal=a;
senal=senal-media;
 senalmod=load('_Modelo_.txt');
 mediamod=mean(senalmod);
 mediamod2=mediamod;
 for i=1:m    %Elaboramos una matriz con las medias para restarla luego.
    mediamod=[mediamod;mediamod2];
 end
 senal=senal+mediamod;

for t=1       %Veces que queremos que se repita el mecanismo de corrección.
 for l=1:78
    maximoind=max(senal(:,l));
    minimoind=min(senal(:,l));
    rango_ind=abs(maximoind-minimoind);
    maximomod=max(senalmod(:,l));
    minimomod=min(senalmod(:,l));
    rango_mod=abs(maximomod-minimomod);
    Rsuperior=maximomod+((rango_mod));%R será el rango a partir del cual
    Rinferior=minimomod-((rango_mod));%cortemos la señal.
    mediamodelo=mean(senalmod(:,l));
        if rango_ind>(15/10)*rango_mod %Si el rango de nuestra señal es x
            l;   %veces mayor que el del modelo lo tratamos, lo corregimos.
            senal(1,l)=senalmod(1,l); %Para situarnos en donde están las
muestras de referencia del modelo.
```

```
            if abs(minimoind-mediamodelo)<abs(maximoind-mediamodelo)
                %disp('mínimo');
                    %Este es el caso de que el minimo de la señal sea más
                    %correcto que el máximo (comparandolo con el modelo).
                anterior=0; %anterior=0 Significa que la muestra anterior
                            %no era parte de un pico, si es =1 es que si.
                for i=2:m
                    if senal(i,l)>(minimoind+abs(rango_ind*(5/10)))
                        if anterior==1
                            senal(i,l)=senal(i,l)-resta;
   %En el caso de que la anterior muestra haya sido incorrecta lo que
   %hacemos es bajar la diferencia porque esta señal tiene su parte buena.
                        else
   %En el caso de encontrarnos la primera muestra
   %del pico mala lo que hacemos es igualarla a la anterior.
                            resta=(senal(i,l)-senal(i-1,l));
                            senal(i,l)=senal(i-1,l);
                            anterior=1;
                        end
                    else
                         senal(i,l)=senal(i,l);
                         anterior=0;
                    end
                end
            else
                %disp('máximo');
                anterior=0;
                for i=2:m
                    if senal(i,l)<(maximoind-abs(rango_ind*(5/10)))
                        if anterior==1
                            senal(i,l)=senal(i,l)-resta;
   %En el caso de que la anterior muestra haya sido incorrecta lo que
   %hacemos es bajar la diferencia porque esta señal tiene su parte buena.
                        else
   %En el caso de encontrarnos la primera muestra del pico mala lo que
   %hacemos es igualarla a la anterior.
                            resta=(senal(i,l)-senal(i-1,l));
                            senal(i,l)=senal(i-1,l);
                            anterior=1;
                        end
                    else
                         senal(i,l)=senal(i,l);
                         anterior=0;
                    end
                end
            end
        end
 end
%%%%%%%%%%%%%%%%Factor que iguala el rango de la señal a la del modelo.
if 3>2  %Se puede activar o no, según nos interese.
  for i=1:78
    w=zeros(100,78);

rango_vector=[(std(senal(1:100,i))),(std(senal(101:200,i))),(std(senal(201:300,i
))),(std(senal(301:400,i)))];
    [minimo,indice]=min(rango_vector);
    w(:,i)=senal(((100*indice)-99):(100*indice),i);
    maximoind=max(w(:,i));
    minimoind=min(w(:,i));
    rango_ind=abs(maximoind-minimoind);
    maximomod=max(senalmod(:,i));
```

```
    minimomod=min(senalmod(:,i));
    rango_mod=abs(maximomod-minimomod);
    factor=rango_mod/rango_ind;
    media=mean(senal(:,i));
    senal(:,i)=senal(:,i)-media;
    senal(:,i)=factor*senal(:,i);
    senal(:,i)=senal(:,i)+media;
  end
end
%%%%%%%%%%%%%%%%%%%%%%%%%
 mediadespues=mean(senal);
 mediadespues2=mediadespues;
 m=size(senal,1)-1;
 for i=1:m %Elaboramos una matriz con las medias para restarla luego.
    mediadespues=[mediadespues;mediadespues2];
 end
 senal=senal-mediadespues;
 senal=senal+mediamod;
end

a=a-media+mediamod;
if 3>4          %Este if es para si queremos dibujar o no.
        datcorr_2=senal;
        datcorr=a;
        figure('Position',[82   185   560   420]);
        for i=1:3:78

            plot(datcorr(:,i),'r');
            hold on
            plot(senalmod(:,i),'g');
            hold on
            plot (datcorr_2(:,i));
            title(['Coordenada X Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            legend('Antes','Modelo','Despues','Location','SouthWest')
            pause
            hold off

            plot(datcorr(:,i+1),'r');
            hold on
            plot(senalmod(:,i+1),'g');
            hold on
            plot (datcorr_2(:,i+1));
            title(['Coordenada Y Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            legend('Antes','Modelo','Despues','Location','SouthWest')
            pause
            hold off

            plot(datcorr(:,i+2),'r');
            hold on
            plot(senalmod(:,i+2),'g');
            hold on
            plot (datcorr_2(:,i+2));
            title(['Coordenada Z Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            legend('Antes','Modelo','Despues','Location','SouthWest')
            pause
            hold off
        end
        delete (2);
```

```
end

%SEGUNDO PASO - Corregimos el resto de señales, los picos y artefactos
%restantes.
%%%%%%%%%%%%%%%%%%%%%%%%PRIMERA PARTE%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%disp('Primera parte');
media=mean(senal);          %Hallamos la media de las columnas.
media2=media;
m=size(senal,1)-1;
for i=1:m    %Elaboramos una matriz con las medias para restarla luego.
    media=[media;media2];
end
matsinmedia=senal-media;    %Restamos a la matriz original las medias.
sigma=std(matsinmedia); %Obtenemos la desviación tìpica por columnas.
rango=(7/4)*sigma;          %Tomamos como rango 3*desviación típica.

disp('Calculando primera pasada');
for columna=1:78
    for fila=2:m
%Ahora vamos comprobando muestra por muestra que no
        if abs(matsinmedia(fila,columna))>rango(columna) &&
abs(matsinmedia(fila+1,columna))<rango(columna) %sobrepasamos el rango fijado.
Si la muestra
            matsinmedia(fila,columna)=(matsinmedia(fila-
1,columna)+matsinmedia(fila+1,columna))/2;          %sobrepasa dicho rango
haremos la media aritmética
        end
%entre las muestras vecinas.

        if abs(matsinmedia(fila,columna))>rango(columna) &&
abs(matsinmedia(fila+1,columna))>rango(columna) %En este caso no es una muestra
la mala sino que hay
            fila2=fila;
%una serie continua de datos erróneos.
            cont=2;
            while abs(matsinmedia(fila2+1,columna))>rango(columna) && fila2<m
            cont=cont+1;
            fila2=fila2+1;
            end
            escalon=(matsinmedia(fila2+1,columna)-matsinmedia(fila-
1,columna))/(cont);
            for i=fila:fila2
                matsinmedia(i,columna)=matsinmedia(i-1,columna)+escalon;
            end
            %matsinmedia(fila,columna)=0; %Para probar que funciona.
        end
    end
end
disp('Final primera pasada');
matrizfinal=matsinmedia+media;%Sumamos la media, obtenemos la matriz buena.
datcorr=matrizfinal;

%%%%%%%%%%%%%%%%%%%%%%%%SEGUNDA PARTE%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('Segunda parte');
b=datcorr;
media_2=mean(b);          %Hallamos la media de las columnas.
media2_2=media_2;
n=size(b,1)-1;
for i=1:n  %Elaboramos una matriz con las medias para restarla luego.
    media_2=[media_2;media2_2];
end
```

```
matsinmedia_2=b-media_2;    %Restamos a la matriz original las medias.
sigma_2=std(matsinmedia_2); %Obtenemos la desviación tìpica por columnas.
rango_2=(12/4)*sigma_2;          %Tomamos como rango 3*desviación típica.

disp('Calculando segunda pasada');
for columna=1:78
    for fila=2:n
%Ahora vamos comprobando muestra por muestra que no
        if abs(matsinmedia_2(fila,columna))>rango_2(columna) &&
abs(matsinmedia_2(fila+1,columna))<rango_2(columna)%sobrepasamos el rango
fijado. Si la muestra
            matsinmedia_2(fila,columna)=(matsinmedia_2(fila-
1,columna)+matsinmedia_2(fila+1,columna))/2;          %sobrepasa dicho rango
haremos la media aritmética
        end
%entre las muestras vecinas.
        if abs(matsinmedia_2(fila,columna))>rango_2(columna) &&
abs(matsinmedia_2(fila+1,columna))>rango_2(columna)%En este caso no es una
muestra la mala sino que hay
            fila2_2=fila;
%una serie continua de datos erróneos.
            cont_2=2;
            while abs(matsinmedia_2(fila2+1,columna))>rango_2(columna) &&
fila2_2<m
            cont_2=cont_2+1;
            fila2_2=fila2_2+1;
            end
            escalon_2=(matsinmedia_2(fila2+1,columna)-matsinmedia_2(fila-
1,columna))/(cont_2);
            for i=fila:fila2_2
                matsinmedia_2(i,columna)=matsinmedia_2(i-1,columna)+escalon_2;
            end
            %matsinmedia(fila,columna)=0;
        end
    end
end

disp('Final segunda pasada');
dinamica=matsinmedia_2;
tamano=size(dinamica);
tamano=tamano(:,1);
matrizfinal_2=matsinmedia_2+media_2;%Sumamos la media, obtenemos la matriz
buena.
save datcorr_antes.txt matrizfinal_2 -ascii; %Guardamos antes de trasladarlo
if archivo == '_Modelo_.txt'        %y corregir para ver bien los artefactos.
    opcion=matrizfinal_2; %No hacemos cambios si es el modelo.
else
    estaticabuena=load('estaticabuena.txt');   %Cargamos el modelo de cuerpo que
está corregido.
    estaticabuena=estaticabuena(1:tamano,:);   %Ajustamos las matrices para
sumarlas puesto que la buena de referencia tiene más muestras.
    opcion=dinamica+estaticabuena;             %Sumamos dinámica al modelo
bueno.
end

datcorr_2=opcion;  %Cargamos los datos originales y los corregidos para
datorig_2=a;       %compararlos en las gráficas.
save datorig_2.txt datorig_2 -ascii;

estatica=mean(datcorr_2); %Hallamos la media de las columnas para calcular
estatica2=estatica;       %la figura estática.
```

```
m_est=size(datcorr_2,1)-1;
n_est=m+1;
for i=1:m_est  %Elaboramos una matriz con las medias para restarla luego.
    estatica=[estatica;estatica2];
end

%Corregimos para enderezar la cadera y el cuerpo en general.
 %Calculamos el error entre los dos huesos de la cadera números 4 y 22.
correcZ=(((estatica(:,4*3)+estatica(:,22*3))/2)-(estatica(:,4*3)));

for i=3:3:57      %Coordenada z corrección (medio cuerpo+columna+cabeza).
    estatica(:,i)=estatica(:,i)+correcZ;
    datcorr_2(:,i)=datcorr_2(:,i)+correcZ;
end

for i=60:3:78      %Coordenada z corrección (el otro medio cuerpo).
    estatica(:,i)=estatica(:,i)-correcZ;
    datcorr_2(:,i)=datcorr_2(:,i)-correcZ;
end
disp('Enderezada la cadera y el cuerpo');

%Guardamos en la carpeta de programa los archivos necesarios.
if archivo == '_Modelo_.txt'
    save estaticabuena.txt estatica -ascii
    disp('Es el modelo');
end
save datcorr.txt datcorr -ascii;
save datcorr_2.txt datcorr_2 -ascii;
save datorig.txt datorig -ascii;
save sinartefactos.txt datcorr_2 -ascii;
save estatica.txt estatica -ascii;
disp('Guardados todos los "txt"');
```

Volver

2.- Código - Menú.

```
%Con el siguiente código se preparan las "figures".

pantalla_tamano=get( 0, 'ScreenSize' );
horizontal=pantalla_tamano(1,3);
vertical=pantalla_tamano(1,4);
tam_hor=(horizontal/2)-120;
tam_vert=tam_hor;
x_izq=50;
x_der=horizontal-tam_hor-60;
y_izq=(vertical-tam_hor)/2;
y_der=y_izq;
izquierda=[x_izq,y_izq,tam_hor,tam_vert];
derecha=[x_der,y_der,tam_hor,tam_vert];

nopiesmodelo=0;
sipiesmodelo=1;
save piesmodelo nopiesmodelo -ascii;
a=load('datcorr_2.txt');
corregir=questdlg('¿Corregir la señal?','Opciones','Si','No','No');
 switch corregir
    case 'Si'
      manual = questdlg('Seleccione la corrección manual:','Opciones','Pies del
Modelo','Corregir todos los sensores','Corregir pies','Corregir pies');
      switch manual
        case 'Pies del Modelo'
                %Sustituir los pies por los del modelo.
                save piesmodelo sipiesmodelo -ascii;
                cd(directorio_programa);
                cd('Datos');
                modelo=load('_Modelo_.txt');
                modelo2=modelo;
                for i=1:10  %Repetimos la señal las veces que queramos.
                modelo=[modelo;modelo2];
                end
                modelo=modelo(1:size(a),:);
                cd ..
                cd('data');
                b=a;    %Solo para la representación ya sustituiremos luego.
                a(:,19)=modelo(:,19);  %De momento mostramos las señales de
                a(:,20)=modelo(:,20);  %nuestro modelo pero más adelante lo
                a(:,21)=modelo(:,21);  %que haremos será mezclar los DPM
                a(:,22)=modelo(:,22);  %para tener tanto la señal del
                a(:,23)=modelo(:,23);  %paciente como la del modelo
                a(:,24)=modelo(:,24);  %sincronizada.
                a(:,73)=modelo(:,73);
                a(:,74)=modelo(:,74);
                a(:,75)=modelo(:,75);
                a(:,76)=modelo(:,76);
                a(:,77)=modelo(:,77);
                a(:,78)=modelo(:,78);
                %Ahora hacemos DPM del modelo para tener los pies
                %preparados para luego.
                datos_buenos=datcorr_2; %Guardamos los datos para dejarlo
                directorio1=pwd;        %todo tal cual estaba.
                cd('DPM');
                save datcorr_2.txt modelo2 -ascii;
                MainModificado;
                datcorr_2=load('corregido.txt');
```

```
                cd(directorio1);
                save DPM_Pies_Modelo_.txt datcorr_2 -ascii;
                datcorr_2=datos_buenos; %Volvemos a los datos de antes.
                save datcorr_2 datcorr_2 -ascii;
        case 'Corregir todos los sensores'
                Paso_1_2_Quitar_picos;
                a=load('datdespues.txt');
                b=a;
        case 'Corregir pies'
                %Corregir las señales SOLO de los pies.
                Paso_1_3_Corregir_1x1;
                a=load('datdespues.txt');
                b=a;

      end
    case 'No'
        b=a;
 end

figure('Position',izquierda)
for  i=1:3:76
     hold on
     plot(a(:,i),'r')
     hold on
     plot(a(:,i+1),'g')
     hold on
     plot(a(:,i+2),'b')
end
a=b; %Como antes, esto es solo para la representación luego sustituiremos.

 casos = questdlg('Seleccione entre las siguientes opciones:','Opciones','Cortar
Señal','Repetir Mejor Segmento','Nada','Cortar Señal');
 switch casos
    case 'Cortar Señal'
        %Cortar la señal y quedarnos con las "x" primeras muestras.
        casos = questdlg('Seleccione hasta que muestra incluir en el
recorte:','Opciones','100-200','250-350','No cortar','100-200');
        switch casos
            case '100-200'
                delete(2)
                caso100 = questdlg('Seleccione hasta que muestra incluir en el
recorte:','Opciones','100','150','200','100');
                switch caso100
                    case '100'
                        a=a(1:100,:);
                    case '150'
                        a=a(1:150,:);
                    case '200'
                        a=a(1:200,:);
                end
            case '250-350'
                delete(2)
                caso250 = questdlg('Seleccione hasta que muestra incluir en el
recorte:','Opciones','250','300','350','250');
                switch caso250
                    case '250'
                        a=a(1:250,:);
                    case '300'
                        a=a(1:300,:);
                    case '350'
                        a=a(1:350,:);
```

```
                end
            case 'No cortar'
                delete(2)
        end
    case 'Repetir Mejor Segmento'
        delete(2)
        Paso_1_1_Parte_buena_senal;
        b1=load('datcorr_2.txt');
        for i=1:20  %Repetimos la señal las veces que queramos.
        a=[b1;a];
        end
     case 'Nada'
         delete(2)
 end

save datcorr_2.txt a -ascii;
datcorr_2=load('datcorr_2.txt');
directorio_data=pwd;
cd('DPM');      %Guardamos en DPM los datos hasta ahora buenos.
save datcorr_2.txt datcorr_2 -ascii;
cd(directorio_data);
```

Volver

3.- Código - Paso_1_1_Parte_buena_senal.

```
a=load('sinartefactos.txt');
save datantes.txt a -ascii;

%PROGRAMA

b=a;
n=size(b);
filas=n(1,1);
columnas=n(1,2);
c=[];
media=mean(b(1:100,:));
for l=1:columnas

rango_vector=[(std(b(1:100,l))),(std(b(101:200,l))),(std(b(201:300,l))),(std(b(3
01:400,l)))];
    [minimo,indice]=min(rango_vector);
    c(:,l)=b(((100*indice)-99):(100*indice),l);
end

save datcorr_2.txt c -ascii,
save datdespues.txt c -ascii;
%Dibujamos los resultados:

j=1; %j=0 DIBUJAMOS   %j=otro NO DIBUJAMOS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%DIBUJAR
if j==0
Dibujar;
end
```

Volver

4.- Código - Paso_1_2_Quitar_picos.

```
disp('Paso_1_1_Quitar_Picos');

    a=load('datcorr_2.txt');
    save datantes.txt a -ascii;

%PROGRAMA

b=a;
n=size(b);
filas=n(1,1);
columnas=n(1,2);
c=[];

%Mismo algoritmo que en el Paso_1_0 pero restricciones más fuertes.
media=mean(a);          %Hallamos la media de las columnas.
media2=media;
m=size(a,1)-1;

for i=1:m        %Elaboramos una matriz con las medias para restarla luego.
    media=[media;media2];
end
senal=b;
senal=senal-media;
 senalmod=load('_Modelo_.txt');
 mediamod=mean(senalmod);
 mediamod2=mediamod;
 for i=1:m       %Elaboramos una matriz con las medias para restarla luego.
    mediamod=[mediamod;mediamod2];
 end
 senal=senal+mediamod;

for l=1:columnas
    maximoind=max(senal(:,l));
    minimoind=min(senal(:,l));
    rango_ind=abs(maximoind-minimoind);
    maximomod=max(senalmod(:,l));
    minimomod=min(senalmod(:,l));
    rango_mod=abs(maximomod-minimomod);
    Rsuperior=maximomod+((rango_mod)); %R es será el rango a partir del
    Rinferior=minimomod-((rango_mod)); %cual cortemos la señal.
    mediamodelo=mean(senalmod(:,l));
        if rango_ind>(12/10)*rango_mod
                %Si el rango de nuestra señal es x veces mayor que el
                %del modelo lo tratamos, lo corregimos.
            senal(1,l)=senalmod(1,l);
      %Para situarnos en donde están las muestras de referencia del modelo.
            if abs(minimoind-mediamodelo)<abs(maximoind-mediamodelo)
                %este es el caso de que ell minimo de la señal sea más
                %correcto que el máximo (comparandolo con el modelo).
                anterior=0;  %anterior=0 Significa que la muestra anterior
                             %no era parte de un pico, si es =1 es que si.
                %disp('minimo');
                for i=2:m
                    if senal(i,l)>(minimoind+abs(rango_ind*(5/10)))
                        if anterior==1
    %En el caso de que la anterior muestra haya sido incorrecta lo que
    %hacemos es bajarla diferencia porque esta señal tiene su parte buena.
```

```
                            senal(i,l)=senal(i,l)-resta;
                        else
    %En el caso de encontrarnos la primera muestra del pico mala lo que
    %hacemos es igualarla a la anterior.
                            resta=(senal(i,l)-senal(i-1,l));
                            senal(i,l)=senal(i-1,l);
                            anterior=1;
                        end
                    else
                         senal(i,l)=senal(i,l);
                         anterior=0;
                    end
                end
            else
                %disp('maximo');
                anterior=0;
                for i=2:m
                    if senal(i,l)<(maximoind-abs(rango_ind*(5/10)))
                    %if 0>1
                        if anterior==1
    %En el caso de que la anterior muestra haya sido incorrecta lo que
    %hacemos es bajar la diferencia porque esta señal tiene su parte buena.
                            senal(i,l)=senal(i,l)-resta;
                        else
    %En el caso de encontrarnos la primera muestra del pico mala lo que
    %hacemos es igualarla a la anterior.
                            resta=(senal(i,l)-senal(i-1,l));
                            senal(i,l)=senal(i-1,l);
                            anterior=1;
                        end
                    else
                         senal(i,l)=senal(i,l);
                         anterior=0;
                    end
                end
            end
        end
end
b=senal;
c=b;
save datdespues.txt c -ascii;
%Dibujamos los resultados:

j=1; %j=0 DIBUJAMOS   %j=otro NO DIBUJAMOS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%DIBUJAR
if j==0
Dibujar;
end
```

Volver

5.- Código - Paso_1_3_Corregir_1x1.

```
disp('Paso_1_3_Corregir_1x1');

    a=load('datcorr_2.txt');
    save datantes.txt a -ascii;

%PROGRAMA

b=a;
n=size(b);
filas=n(1,1);
c=[];

%Mismo algoritmo que en el Paso_1_0 pero para los pies.
media=mean(a);          %Hallamos la media de las columnas.
media2=media;
m=size(a,1)-1;

for i=1:m %Elaboramos una matriz con las medias para restarla luego.
    media=[media;media2];
end
senal=b;
senal=senal-media;
 senalmod=load('_Modelo_.txt');
 mediamod=mean(senalmod);
 mediamod2=mediamod;
 for i=1:m %Elaboramos una matriz con las medias para restarla luego.
    mediamod=[mediamod;mediamod2];
 end
 senal=senal+mediamod;

for l=19:24
    maximoind=max(senal(:,l));
    minimoind=min(senal(:,l));
    rango_ind=abs(maximoind-minimoind);
    maximomod=max(senalmod(:,l));
    minimomod=min(senalmod(:,l));
    rango_mod=abs(maximomod-minimomod);
    Rsuperior=maximomod+((rango_mod));  %R es será el rango a partir del
    Rinferior=minimomod-((rango_mod));  %cual cortemos la señal.
    mediamodelo=mean(senalmod(:,l));
        if rango_ind>(12/10)*rango_mod
           %Si el rango de nuestra señal es x veces mayor que el
           %del modelo lo tratamos, lo corregimos.
            senal(1,l)=senalmod(1,l);
  %Para situarnos en donde están las muestras de referencia del modelo.
            if abs(minimoind-mediamodelo)<abs(maximoind-mediamodelo)
       %este es el caso de que ell minimo de la señal sea más correcto que
       %el máximo (comparandolo con el modelo)
                anterior=0; %anterior=0 Significa que la muestra anterior
                            %no era parte de un pico, si es =1 es que si.
                %disp('minimo');
                for i=2:m
                    if senal(i,l)>(minimoind+abs(rango_ind*(5/10)))
                        if anterior==1
    %En el caso de que la anterior muestra haya sido incorrecta lo que
    %hacemos es bajar la diferencia porque esta señal tiene su parte buena.
                            senal(i,l)=senal(i,l)-resta;
```

```
                        else
    %En el caso de encontrarnos la primera muestra del pico mala lo que
    %hacemos es igualarla a la anterior.
                            resta=(senal(i,l)-senal(i-1,l));
                            senal(i,l)=senal(i-1,l);
                            anterior=1;
                        end
                    else
                         senal(i,l)=senal(i,l);
                         anterior=0;
                    end
                end
            else
                %disp('maximo');
                anterior=0;
                for i=2:m
                    if senal(i,l)<(maximoind-abs(rango_ind*(5/10)))
                    %if 0>1
                        if anterior==1
    %En el caso de que la anterior muestra haya sido incorrecta lo que
    %hacemos es bajar la diferencia porque esta señal tiene su parte buena.
                            senal(i,l)=senal(i,l)-resta;
                        else
    %En el caso de encontrarnos la primera muestra del pico mala lo que
    %hacemos es igualarla a la anterior.
                            resta=(senal(i,l)-senal(i-1,l));
                            senal(i,l)=senal(i-1,l);
                            anterior=1;
                        end
                    else
                         senal(i,l)=senal(i,l);
                         anterior=0;
                    end
                end
            end
        end
 b=senal;

 %Dibujamos los resultados:

figure('Position',izquierda)
hold on
plot(a(:,l),'r')
hold on
plot(b(:,l),'b')
title(['Señal ',int2str(floor(l))],'fontsize',25,'Color','b');
legend('Antes','Despues','Location','SouthWest')
c=a;
 sustituir=questdlg('¿Desea sustituir la señal?','Opciones','Si','No','No');
 switch sustituir
     case 'Si'
        c(:,l)=b(:,l);
     case 'No'
 end
 delete(2)
end


for l=73:78

    maximoind=max(senal(:,l));
```

```
 minimoind=min(senal(:,l));
 rango_ind=abs(maximoind-minimoind);
 maximomod=max(senalmod(:,l));
 minimomod=min(senalmod(:,l));
 rango_mod=abs(maximomod-minimomod);
 Rsuperior=maximomod+((rango_mod));  %R es será el rango a partir del
 Rinferior=minimomod-((rango_mod));  %cual cortemos la señal.
 mediamodelo=mean(senalmod(:,l));
     if rango_ind>(12/10)*rango_mod
         %Si el rango de nuestra señal es x veces mayor que
         %el del modelo lo tratamos, lo corregimos.
         senal(1,l)=senalmod(1,l);
%Para situarnos en donde están las muestras de referencia del modelo.
         if abs(minimoind-mediamodelo)<abs(maximoind-mediamodelo)
    %este es el caso de que ell minimo de la señal sea más correcto que
    %el máximo (comparandolo con el modelo)
             anterior=0; %anterior=0 Significa que la muestra anterior
                         %no era parte de un pico, si es =1 es que si.
             %disp('minimo');
             for i=2:m
                 if senal(i,l)>(minimoind+abs(rango_ind*(5/10)))
                     if anterior==1
 %En el caso de que la anterior muestra haya sido incorrecta lo que
 %hacemos es bajar la diferencia porque esta señal tiene su parte buena.
                         senal(i,l)=senal(i,l)-resta;
                     else
 %En el caso de encontrarnos la primera muestra del pico mala lo que
 %hacemos es igualarla a la anterior.
                         resta=(senal(i,l)-senal(i-1,l));
                         senal(i,l)=senal(i-1,l);
                         anterior=1;
                     end
                 else
                      senal(i,l)=senal(i,l);
                      anterior=0;
                 end
             end
         else
             %disp('maximo');
             anterior=0;
             for i=2:m
                 if senal(i,l)<(maximoind-abs(rango_ind*(5/10)))
                 %if 0>1
                     if anterior==1
 %En el caso de que la anterior muestra haya sido incorrecta lo que
 %hacemos es bajar la diferencia porque esta señal tiene su parte buena.
                         senal(i,l)=senal(i,l)-resta;
                     else
 %En el caso de encontrarnos la primera muestra del pico mala lo que
 %hacemos es igualarla a la anterior.
                         resta=(senal(i,l)-senal(i-1,l));
                         senal(i,l)=senal(i-1,l);
                         anterior=1;
                     end
                 else
                      senal(i,l)=senal(i,l);
                      anterior=0;
                 end
             end
         end
     end
```

```
 b=senal;

%Dibujamos los resultados:

figure('Position',izquierda)
hold on
plot(a(:,l),'r')
hold on
plot(b(:,l),'b')
title(['Señal ',int2str(floor(l))],'fontsize',25,'Color','b');
legend('Antes','Despues','Location','SouthWest')
c=a;
 sustituir=questdlg('¿Desea sustituir la señal?','Opciones','Si','No','No');
 switch sustituir
     case 'Si'
        c(:,l)=b(:,l);
     case 'No'
 end
 delete(2)
end
save datdespues.txt c -ascii;
```

Volver

6.- Código - Paso_2_Centrar_cuerpo.

```
%Centramos el cuerpo entero (26 sensores en el centro de nuestras gráficas
%tomando como referencia el sensor 18 (final de la columna vertebral).
%Corregimos la altura, coordenada Y.

disp('Paso_2_Centrar_cuerpo')

datcorr_2=load('DPM.txt');
estatica=mean(datcorr_2);
estatica2=estatica;
m_est=size(datcorr_2,1)-1;
datorig=load('datorig.txt');
estaticaindividuo=mean(datorig);
estaticaindividuo2=estaticaindividuo;
for i=1:m_est
    estatica=[estatica;estatica2];
    estaticaindividuo=[estaticaindividuo;estaticaindividuo2];
end

%Centramos todas las coordenadas.

CentraY=(((estatica(1,3*8-1)+estatica(1,3*26-1))/2));
%Calculamos las alturas de cabeza a pies del modelo y el individuo.
hY=CentraY;
CentraYind=(((estaticaindividuo(1,3*8-1)+estaticaindividuo(1,3*26-1))/2));
hYind=CentraYind;

for i=2:3:77      %coordenada y corrección.
    estatica(:,i)=hY-estatica(:,i);
    estaticaindividuo(:,i)=hYind-estaticaindividuo(:,i);
    datcorr_2(:,i)=hY-datcorr_2(:,i);
end

%calculamos la media de todos los puntos de la columna (del 12 al 18).
columnavertebral=(estatica(1,3*12-2)+estatica(1,3*13-2)+estatica(1,3*14-
2)+estatica(1,3*15-2)+estatica(1,3*16-2)+estatica(1,3*17-2)+estatica(1,3*18-
2))/7;
CentraX=(columnavertebral);
hX=CentraX;
columnavertebralind=(estaticaindividuo(1,3*12-2)+estaticaindividuo(1,3*13-
2)+estaticaindividuo(1,3*14-2)+estaticaindividuo(1,3*15-
2)+estaticaindividuo(1,3*16-2)+estaticaindividuo(1,3*17-
2)+estaticaindividuo(1,3*18-2))/7;
CentraXind=(columnavertebralind);
hXind=CentraXind;

for i=1:3:76      %coordenada x corrección.
    estatica(:,i)=hX-estatica(:,i);
    estaticaindividuo(:,i)=hXind-estaticaindividuo(:,i);
    datcorr_2(:,i)=hX-datcorr_2(:,i);
end
if archivo == '_Modelo_.txt'
   save estaticamodelo.txt estatica -ascii
   %disp('Guardado estaticamodelo');
end
save estaticaindividuo.txt estaticaindividuo -ascii;
save estatica.txt estatica -ascii;
save datcorr_2.txt datcorr_2 -ascii;
```

```
save DPM.txt datcorr_2 -ascii;
disp('Cuerpo centrado');
```

Volver

7.- Código - Paso_3_Factor_ensanchamiento_y_altura.

```
%Como estamos tomando la estatica del modelo y no tienen porque medir ni
%de altura ni de anchura lo mismo las dos personas, aplicaremos dicho
%factor de correción.
%Restamos a los datos la estatica que teníamos para despues de
%este programa tenerlos con la estática buena.

disp('Paso_3_Factor_ensanchamiento_y_altura')

estatica=load('estatica.txt');
datcorr_2=load('DPM.txt');
datorig_2=load('datorig.txt');

datcorr_2=datcorr_2-estatica;
a=load('estaticamodelo.txt');   %MODELO
modelo=a;
b=load('estaticaindividuo.txt'); %INDIVIDUO
individuo=b;

        %Calculamos los puntos de referencia
ptorefXmodelo=    ((a(1,3*21-2)+a(1,3*3-2))/2);
%Para la coord. X comparo las medias de las columnas que son más fiables.
ptorefXindividuo=((b(1,3*21-2)+b(1,3*3-2))/2);
ptorefYmodelo=    ((a(1,3*21-1)+a(1,3*3-1))/2);
%Punto de entre los reflectantes de la cadera que servirá de referencia.
ptorefYindividuo=((b(1,3*21-1)+b(1,3*3-1))/2);

        %Calculamos el factor de multiplicación
disp('Factores de multiplicación');
distanciamodelo=    (a(1,3*3-2)-ptorefXmodelo);
%Calculamos la distancia del punto de referencia al sensor 3 de la cadera.
distanciaindividuo=(b(1,3*3-2)-ptorefXindividuo);
factorX=(distanciaindividuo/distanciamodelo)
alturamodelo=    (a(1,3*10-1)-ptorefYmodelo);
%Calculamos la altura desde el pto de ref. al sensor 10 de la cabeza.
alturaindividuo=(b(1,3*10-1)-ptorefYindividuo);
factorY=(alturaindividuo/alturamodelo)

%Preparamos los modelos con los puntos de referencia para tomar el
%punto de referencia como si fuera el 0.
        %Arreglo de la coordenada X.
for i=1:3:76
    %a(:,i)=a(:,i)-ptorefXmodelo; %Colocamos el individuo en el punto de
    %referencia %No es necesario puesto que lo tenemos centrado en 0.
    a(:,i)=factorX*a(:,i);
    %a(:,i)=a(:,i)+ptorefXmodelo;
    %Devolvemos el individuo a su correspondiente situación.
end

        %Arreglo de la coordenada Y.
for i=2:3:77
    a(:,i)=a(:,i)-ptorefYmodelo;
    a(:,i)=factorY*a(:,i);              %Si nos fijamos al hacer el cambio
    %del muñeco que estaba boca abajo ahora el factorY es negativo y así
    %sale todo bien ya después!
    a(:,i)=a(:,i)+(ptorefYmodelo*factorY);
    %Volvemos a poner los modelos en sus verdaderas coordenadas.
end
```

```
estatica=a;
save estatica.txt a -ascii;
datcorr_2=datcorr_2+estatica;
save datcorr_2.txt datcorr_2 -ascii;
resultado=a;

x=0;%Cambiar para ver o no las gráficas. x=0 no se ven, con x=1 se ven.

if x>0
       figure(2)
       for q=1:1;
            j=1:26;
            r=j-1;
            plot(modelo(q,3*r+1),(modelo(q,3*r+2)),'r.');
            hold on
            %plot(individuo(q,3*r+1),(individuo(q,3*r+2)),'g.');
            %hold on
            plot(resultado(q,3*r+1),(resultado(q,3*r+2)),'b.');
            %legend('Modelo','Individuo','Resultado','Location','Best')
            legend('Modelo','Resultado','Location','Best')
            axis([-500 500 -100 2100]);
            grid on
            %title('Modelo Modificado','fontsize',25,'Color','b');
            Xlabel('LEFT-RIGHT');
            Ylabel('UP-DOWN');
       end
       disp('Presiona ENTER para continuar');
       pause
       delete(2);
end
```

Volver

8.- Código - Dibujar.

```
%Programa para centrar las gráficas bien en la pantalla.Así podemos ver las
%dos a la vez.

pantalla_tamano=get( 0, 'ScreenSize' );
horizontal=pantalla_tamano(1,3);
vertical=pantalla_tamano(1,4);
tam_hor=(horizontal/2)-120;
tam_vert=tam_hor;
x_izq=50;
x_der=horizontal-tam_hor-60;
y_izq=(vertical-tam_hor)/2;
y_der=y_izq;
izquierda=[x_izq,y_izq,tam_hor,tam_vert];
derecha=[x_der,y_der,tam_hor,tam_vert];

        datantes=load('datantes.txt');
        datdespues=load('datdespues.txt');

%x=0 Ver las gráficas separadas.
%x=otro Ver las gráficas superpuestas.
x=1;

long=size(a);
long=long(1,2)-1;
if x==0
        for i=1:3:long

            figure('Position',izquierda)      %(position',[x y w h])
            plot(datantes(:,i),'r');          %Coordenada X del Sensor i
            legend('Antes','Location','SouthWest')
            title(['Coordenada X Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            figure('Position',derecha)
            plot(datdespues(:,i),'g');
            legend('Despues','Location','SouthWest')
            title(['Coordenada X Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause
            delete (1);
            delete (2);

            figure('Position',izquierda)
            plot(datantes(:,i+1),'r');        %Coordenada Y del Sensor i
            legend('Antes','Location','SouthWest')
            title(['Coordenada Y Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            figure('Position',derecha)
            plot(datdespues(:,i+1),'g');
            legend('Despues','Location','SouthWest')
            title(['Coordenada Y Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause
            delete (1);
            delete (2);

            figure('Position',izquierda)
            plot(datantes(:,i+2),'r');        %Coordenada Z del Sensor i
```

```
            legend('Antes','Location','SouthWest')
            title(['Coordenada Z Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            figure('Position',derecha)
            plot(datdespues(:,i+2),'g');
            legend('Despues','Location','SouthWest')
            title(['Coordenada Z Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause
            delete (1);
            delete (2);
        end
else
        for i=1:3:long

            figure('Position',izquierda)
            plot(datantes(:,i),'r');          %Coordenada X del Sensor i
            hold on
            plot(datdespues(:,i),'g');
            legend('Antes','Despues','Location','SouthWest')
            title(['Coordenada X Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause
            hold off
            delete (2);

            figure('Position',izquierda)
            plot(datantes(:,i+1),'r');        %Coordenada Y del Sensor i
            hold on
            plot(datdespues(:,i+1),'g');
            legend('Antes','Despues','Location','SouthWest')
            title(['Coordenada Y Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause
            hold off
            delete (2);

            figure('Position',izquierda)
            plot(datantes(:,i+2),'r');        %Coordenada Z del Sensor i
            hold on
            plot(datdespues(:,i+2),'g');
            legend('Antes','Despues','Location','SouthWest')
            title(['Coordenada Z Sensor
',int2str(floor(1+(i/3)))],'fontsize',25,'Color','b');
            pause
            hold off
            delete (2);
        end
end
```

Volver

9.- Código - Columna.

```
%%%%%%%%%%%%%COMPARAR COLUMNA SEGÚN EL PESO
directorio_principal=pwd;
cd ..
cd('Datos');
modelo=load('_Modelo_.txt');
cd ..
cd('Datos Corregidos DPM')
directorio_datos=pwd;

%1°Cargamos los archivos.


cd(directorio_datos);

if 3>2 %Para anular el meter los archivos manualmente.
aviso1=questdlg('Seleccione el archivo de peso 0','','Ok','Ok');
switch aviso1
    case 'Ok'
        [archivopeso0,directoriocol] = uigetfile('*.*','Seleccione el archivo
correspondiente');
        cd(directoriocol);
        peso0=load(archivopeso0);
        cd(directorio_datos);
    case 'Cancelar'
        return
end

aviso2=questdlg('Seleccione el archivo de peso 2','','Ok','Ok');
switch aviso2
    case 'Ok'
        [archivopeso2,directoriocol2] = uigetfile('*.*','Seleccione el archivo
correspondiente');
        cd(directoriocol2);
        peso2=load(archivopeso2);
        cd(directorio_datos);
    case 'Cancelar'
        return
end

aviso3=questdlg('Seleccione el archivo de peso 4','','Ok','Ok');
switch aviso3
    case 'Ok'
        [archivopeso4,directoriocol4] = uigetfile('*.*','Seleccione el archivo
correspondiente');
        cd(directoriocol4);
        peso4=load(archivopeso4);
        cd(directorio_datos);
    case 'Cancelar'
        return
end

aviso4=questdlg('Seleccione el archivo de peso 6','','Ok','Ok');
switch aviso4
    case 'Ok'
        [archivopeso6,directoriocol6] = uigetfile('*.*','Seleccione el archivo
correspondiente');
        cd(directoriocol6);
```

```
        peso6=load(archivopeso6);
        cd(directorio_datos);
    case 'Cancelar'
        return
end

aviso5=questdlg('Seleccione el archivo de peso 8','','Ok','Ok');
switch aviso5
    case 'Ok'
        [archivopeso8,directoriocol8] = uigetfile('*.*','Seleccione el archivo
correspondiente');
        cd(directoriocol8);
        peso8=load(archivopeso8);
        cd(directorio_datos);
    case 'Cancelar'
        return
end

aviso6=questdlg('Seleccione el archivo de peso 10','','Ok','Ok');
switch aviso6
    case 'Ok'
        [archivopeso10,directoriocol10] = uigetfile('*.*','Seleccione el archivo
correspondiente');
    cd(directoriocol10);
    peso10=load(archivopeso10);
    cd(directorio_datos);
    case 'Cancelar'
        return
end

aviso7=questdlg('Seleccione el archivo de peso 15','','Ok','Ok');
switch aviso7
    case 'Ok'
        [archivopeso15,directoriocol15] = uigetfile('*.*','Seleccione el archivo
correspondiente');
        cd(directoriocol15);
        peso15=load(archivopeso15);
        cd(directorio_datos);
    case 'Cancelar'
        return
end
end

%peso0=load('m5_00_13.txt');peso2=load('m5_02_13.txt');peso4=load('m5_04_13.txt'
);peso6=load('m5_06_13.txt');
%peso8=load('m5_08_13.txt');peso10=load('m5_10_13.txt');peso15=load('m5_15_13.tx
t');

cd(directorio_principal);
cd ..

%2° Comprobamos que los tamaños son suficientes y los ajustamos.

vector_tam=[size(modelo,1),size(peso0,1),size(peso2,1),size(peso4,1),size(peso6,
1),size(peso8,1),size(peso10,1),size(peso15,1)];
minimo_tam=min(vector_tam);
modelo=modelo(1:minimo_tam,1:78);
peso0=peso0(1:minimo_tam,1:78);
peso2=peso2(1:minimo_tam,1:78);
peso4=peso4(1:minimo_tam,1:78);
peso6=peso6(1:minimo_tam,1:78);
```

```
peso8=peso8(1:minimo_tam,1:78);
peso10=peso10(1:minimo_tam,1:78);
peso15=peso15(1:minimo_tam,1:78);

%3°Calculamos las estáticas de todos los pesos.%Nos interesan los puntos
%del 9 al 18 incluidos, es decir, del 25 a 54.

media_modelo=(mean(modelo)); media2_modelo=media_modelo;
media_peso0=(mean(peso0)); media2_peso0=media_peso0;
media_peso2=(mean(peso2)); media2_peso2=media_peso2;
media_peso4=(mean(peso4)); media2_peso4=media_peso4;
media_peso6=(mean(peso6)); media2_peso6=media_peso6;
media_peso8=(mean(peso8)); media2_peso8=media_peso8;
media_peso10=(mean(peso10)); media2_peso10=media_peso10;
media_peso15=(mean(peso15)); media2_peso15=media_peso15;

m=size(modelo,1)-1;
for i=1:m
    media_modelo=[media_modelo;media2_modelo];
    media_peso0=[media_peso0;media2_peso0]; %Obtenemos la media repetida
    media_peso2=[media_peso2;media2_peso2]; %para formar una matriz y luego
    media_peso4=[media_peso4;media2_peso4]; %poder restarla.
    media_peso6=[media_peso6;media2_peso6];
    media_peso8=[media_peso8;media2_peso8];
    media_peso10=[media_peso10;media2_peso10];
    media_peso15=[media_peso15;media2_peso15];
end

%4° Multiplicamos por los factores de dimensionamiento correctos y
%calculamos el resultado final.

estatura_modelo=(media_modelo(1,3*10-1))-(media_modelo(1,3*18-1));
estatura_individuo=(media_peso0(1,3*10-1))-(media_peso0(1,3*18-1));
factor=(estatura_individuo/estatura_modelo);

estatica_modelo=mean(media_modelo);
estatica_peso0=mean(peso0-media_peso0+media_modelo); %Resultado= media de(
estatica_peso2=mean(peso2-media_peso2+media_modelo); %Señal paciente -
estatica_peso4=mean(peso4-media_peso4+media_modelo); %media del paciente +
estatica_peso6=mean(peso6-media_peso6+media_modelo); %estatica del modelo )
estatica_peso8=mean(peso8-media_peso8+media_modelo);
estatica_peso10=mean(peso10-media_peso10+media_modelo);
estatica_peso15=mean(peso15-media_peso15+media_modelo);

dinamica_modelo=modelo-media_modelo;
dinamica_peso0=peso0-media_peso0;
dinamica_peso2=peso2-media_peso2;
dinamica_peso4=peso4-media_peso4;
dinamica_peso6=peso6-media_peso6;
dinamica_peso8=peso8-media_peso8;
dinamica_peso10=peso10-media_peso10;
dinamica_peso15=peso15-media_peso15;

peso0=estatica_peso0;
peso2=estatica_peso2;
peso4=estatica_peso4;
peso6=estatica_peso6;
peso8=estatica_peso8;
peso10=estatica_peso10;
peso15=estatica_peso15;
```

```
%5°Centramos donde nos interesan los datos.

for i=25:3:52       %Coordenada x corrección
    media_modelo(:,i)=media_modelo(1,3*18-2)-media_modelo(:,i);
    peso0(:,i)=peso0(1,3*18-2)-peso0(:,i)+400;
    peso2(:,i)=peso2(1,3*18-2)-peso2(:,i)+800;
    peso4(:,i)=peso4(1,3*18-2)-peso4(:,i)+1200;
    peso6(:,i)=peso6(1,3*18-2)-peso6(:,i)+1600;
    peso8(:,i)=peso8(1,3*18-2)-peso8(:,i)+2000;
    peso10(:,i)=peso10(1,3*18-2)-peso10(:,i)+2400;
    peso15(:,i)=peso15(1,3*18-2)-peso15(:,i)+2800;
end

for i=26:3:53       %Coordenada y corrección
    media_modelo(:,i)=media_modelo(1,3*18-1)-media_modelo(:,i);
    peso0(:,i)=(peso0(1,3*18-1)-peso0(:,i));
    peso2(:,i)=(peso2(1,3*18-1)-peso2(:,i));
    peso4(:,i)=(peso4(1,3*18-1)-peso4(:,i));
    peso6(:,i)=(peso6(1,3*18-1)-peso6(:,i));
    peso8(:,i)=(peso8(1,3*18-1)-peso8(:,i));
    peso10(:,i)=(peso10(1,3*18-1)-peso10(:,i));
    peso15(:,i)=(peso15(1,3*18-1)-peso15(:,i));
end

for i=27:3:54       %Coordenada z corrección
    media_modelo(:,i)=media_modelo(1,3*18)-media_modelo(:,i);
    peso0(:,i)=(peso0(1,3*18)-peso0(:,i));
    peso2(:,i)=(peso2(1,3*18)-peso2(:,i));
    peso4(:,i)=(peso4(1,3*18)-peso4(:,i));
    peso6(:,i)=(peso6(1,3*18)-peso6(:,i));
    peso8(:,i)=(peso8(1,3*18)-peso8(:,i));
    peso10(:,i)=(peso10(1,3*18)-peso10(:,i));
    peso15(:,i)=(peso15(1,3*18)-peso15(:,i));
end

a=1;
b=peso0(1,3*18-1); %Punto de la rodilla.
c=peso0(1,3*10-1); %Punto de la cabeza.
if b>c  %Consiste en darle la vuelta si esta del revés.
    a=-1;
end
factor=abs(factor);
for i=2:3:77
peso0(:,i)=a*factor*peso0(:,i);
peso2(:,i)=a*factor*peso2(:,i);
peso4(:,i)=a*factor*peso4(:,i);
peso6(:,i)=a*factor*peso6(:,i);
peso8(:,i)=a*factor*peso8(:,i);
peso10(:,i)=a*factor*peso10(:,i);
peso15(:,i)=a*factor*peso15(:,i);
end

%Calculamos para representar los rangos de movimiento de los
%diferentes puntos.

maxmodelo=zeros(1,78);minmodelo=maxmodelo;maxpeso0=maxmodelo;
minpeso0=maxmodelo;maxpeso2=maxmodelo;minpeso2=maxmodelo;
maxpeso4=maxmodelo;minpeso4=maxmodelo;maxpeso6=maxmodelo;
minpeso6=maxmodelo;maxpeso8=maxmodelo;minpeso8=maxmodelo;
maxpeso10=maxmodelo;minpeso10=maxmodelo;maxpeso15=maxmodelo;
```

```
minpeso15=maxmodelo;

for i=1:3:78
maxmodelo(1,i)=max(dinamica_modelo(:,i));
minmodelo(1,i)=min(dinamica_modelo(:,i));
maxpeso0(1,i)=max(dinamica_peso0(:,i));
minpeso0(1,i)=min(dinamica_peso0(:,i));
maxpeso2(1,i)=max(dinamica_peso2(:,i));
minpeso2(1,i)=min(dinamica_peso2(:,i));
maxpeso4(1,i)=max(dinamica_peso4(:,i));
minpeso4(1,i)=min(dinamica_peso4(:,i));
maxpeso6(1,i)=max(dinamica_peso6(:,i));
minpeso6(1,i)=min(dinamica_peso6(:,i));
maxpeso8(1,i)=max(dinamica_peso8(:,i));
minpeso8(1,i)=min(dinamica_peso8(:,i));
maxpeso10(1,i)=max(dinamica_peso10(:,i));
minpeso10(1,i)=min(dinamica_peso10(:,i));
maxpeso15(1,i)=max(dinamica_peso15(:,i));
minpeso15(1,i)=min(dinamica_peso15(:,i));
end

maxmodelo=maxmodelo+media_modelo(1,:);
minmodelo=minmodelo+media_modelo(1,:);
maxpeso0=maxpeso0+peso0(1,:);
minpeso0=minpeso0+peso0(1,:);
maxpeso2=maxpeso2+peso2(1,:);
minpeso2=minpeso2+peso2(1,:);
maxpeso4=maxpeso4+peso4(1,:);
minpeso4=minpeso4+peso4(1,:);
maxpeso6=maxpeso6+peso6(1,:);
minpeso6=minpeso6+peso6(1,:);
maxpeso8=maxpeso8+peso8(1,:);
minpeso8=minpeso8+peso8(1,:);
maxpeso10=maxpeso10+peso10(1,:);
minpeso10=minpeso10+peso10(1,:);
maxpeso15=maxpeso15+peso15(1,:);
minpeso15=minpeso15+peso15(1,:);

%6°Dibujamos.

figure('Position',[85 109 1185 511]);
axis([-400 3600 -100 800]);
grid on
%Dibujamos primero la estatica.
for i=9:1:18
hold on
plot(media_modelo(1,3*i-2),media_modelo(1,3*i-
1),'blued','MarkerFaceColor','b','MarkerSize',8)
hold on
plot(peso0(1,3*i-2),peso0(1,3*i-
1),'green.','MarkerFaceColor','g','MarkerSize',25)
hold on
plot(peso2(1,3*i-2),peso2(1,3*i-1),'r.','MarkerFaceColor','r','MarkerSize',25)
hold on
plot(peso4(1,3*i-2),peso4(1,3*i-
1),'cyan.','MarkerFaceColor','c','MarkerSize',25)
hold on
plot(peso6(1,3*i-2),peso6(1,3*i-
1),'magenta.','MarkerFaceColor','magenta','MarkerSize',25)
hold on
plot(peso8(1,3*i-2),peso8(1,3*i-
```

```
1),'yellow.','MarkerFaceColor','y','MarkerSize',25)
hold on
plot(peso10(1,3*i-2),peso10(1,3*i-
1),'black.','MarkerFaceColor','black','MarkerSize',25)
hold on
plot(peso15(1,3*i-2),peso15(1,3*i-
1),'blue.','MarkerFaceColor','blue','MarkerSize',25)
legend('Modelo','Peso 0','Peso 2','Peso 4','Peso 6','Peso 8','Peso 10','Peso
15','Location','Best')
end
%Dibujamos ahora los rangos
for i=9:1:18
hold on
plot(maxmodelo(1,3*i-2),maxmodelo(1,3*i-1),'bluex-')
hold on
plot(minmodelo(1,3*i-2),minmodelo(1,3*i-1),'bluex-')
hold on
plot(maxpeso0(1,3*i-2),maxpeso0(1,3*i-1),'greenx-')
hold on
plot(minpeso0(1,3*i-2),minpeso0(1,3*i-1),'greenx-')
hold on
plot(maxpeso2(1,3*i-2),maxpeso2(1,3*i-1),'rx-')
hold on
plot(minpeso2(1,3*i-2),minpeso2(1,3*i-1),'rx-')
hold on
plot(maxpeso4(1,3*i-2),maxpeso4(1,3*i-1),'cyanx-')
hold on
plot(minpeso4(1,3*i-2),minpeso4(1,3*i-1),'cyanx-')
hold on
plot(maxpeso6(1,3*i-2),maxpeso6(1,3*i-1),'magentax-')
hold on
plot(minpeso6(1,3*i-2),minpeso6(1,3*i-1),'magentax-')
hold on
plot(maxpeso8(1,3*i-2),maxpeso8(1,3*i-1),'yellowx-')
hold on
plot(minpeso8(1,3*i-2),minpeso8(1,3*i-1),'yellowx-')
hold on
plot(maxpeso10(1,3*i-2),maxpeso10(1,3*i-1),'blackx-')
hold on
plot(minpeso10(1,3*i-2),minpeso10(1,3*i-1),'--blackx')
hold on
plot(maxpeso15(1,3*i-2),maxpeso15(1,3*i-1),'--bluex')
hold on
plot(minpeso15(1,3*i-2),minpeso15(1,3*i-1),'--bluex')
set(gca,'LineStyleOrder', '-*|:|o')
end

continuar=questdlg('¿Continuar?','','Ok','Ok');
delete(1)

%7°Centramos donde nos interesan los datos para dibujar ahora de perfil.

for i=25:3:52       %Coordenada x corrección
    media_modelo(:,i)=media_modelo(1,3*18-2)-media_modelo(:,i);
    peso0(:,i)=peso0(1,3*18-2)-peso0(:,i)-400;
    peso2(:,i)=peso2(1,3*18-2)-peso2(:,i)-800;
    peso4(:,i)=peso4(1,3*18-2)-peso4(:,i)-1200;
    peso6(:,i)=peso6(1,3*18-2)-peso6(:,i)-1600;
    peso8(:,i)=peso8(1,3*18-2)-peso8(:,i)-2000;
    peso10(:,i)=peso10(1,3*18-2)-peso10(:,i)-2400;
    peso15(:,i)=peso15(1,3*18-2)-peso15(:,i)-2800;
```

```
end

for i=27:3:54       %Coordenada z corrección
    media_modelo(:,i)=media_modelo(1,3*18)-media_modelo(:,i);
    peso0(:,i)=(peso0(1,3*18)-peso0(:,i))+400;
    peso2(:,i)=(peso2(1,3*18)-peso2(:,i))+800;
    peso4(:,i)=(peso4(1,3*18)-peso4(:,i))+1200;
    peso6(:,i)=(peso6(1,3*18)-peso6(:,i))+1600;
    peso8(:,i)=(peso8(1,3*18)-peso8(:,i))+2000;
    peso10(:,i)=(peso10(1,3*18)-peso10(:,i))+2400;
    peso15(:,i)=(peso15(1,3*18)-peso15(:,i))+2800;
end

%Recalculamos para representar los rangos de movimiento de los
%diferentes puntos que ahora tienen otras coordenadas.

maxmodelo=zeros(1,78);minmodelo=maxmodelo;maxpeso0=maxmodelo;
minpeso0=maxmodelo;maxpeso2=maxmodelo;minpeso2=maxmodelo;
maxpeso4=maxmodelo;minpeso4=maxmodelo;maxpeso6=maxmodelo;
minpeso6=maxmodelo;maxpeso8=maxmodelo;minpeso8=maxmodelo;
maxpeso10=maxmodelo;minpeso10=maxmodelo;maxpeso15=maxmodelo;
minpeso15=maxmodelo;

for i=3:3:78
maxmodelo(1,i)=max(dinamica_modelo(:,i));
minmodelo(1,i)=min(dinamica_modelo(:,i));
maxpeso0(1,i)=max(dinamica_peso0(:,i));
minpeso0(1,i)=min(dinamica_peso0(:,i));
maxpeso2(1,i)=max(dinamica_peso2(:,i));
minpeso2(1,i)=min(dinamica_peso2(:,i));
maxpeso4(1,i)=max(dinamica_peso4(:,i));
minpeso4(1,i)=min(dinamica_peso4(:,i));
maxpeso6(1,i)=max(dinamica_peso6(:,i));
minpeso6(1,i)=min(dinamica_peso6(:,i));
maxpeso8(1,i)=max(dinamica_peso8(:,i));
minpeso8(1,i)=min(dinamica_peso8(:,i));
maxpeso10(1,i)=max(dinamica_peso10(:,i));
minpeso10(1,i)=min(dinamica_peso10(:,i));
maxpeso15(1,i)=max(dinamica_peso15(:,i));
minpeso15(1,i)=min(dinamica_peso15(:,i));
end

maxmodelo=maxmodelo+media_modelo(1,:);
minmodelo=minmodelo+media_modelo(1,:);
maxpeso0=maxpeso0+peso0(1,:);
minpeso0=minpeso0+peso0(1,:);
maxpeso2=maxpeso2+peso2(1,:);
minpeso2=minpeso2+peso2(1,:);
maxpeso4=maxpeso4+peso4(1,:);
minpeso4=minpeso4+peso4(1,:);
maxpeso6=maxpeso6+peso6(1,:);
minpeso6=minpeso6+peso6(1,:);
maxpeso8=maxpeso8+peso8(1,:);
minpeso8=minpeso8+peso8(1,:);
maxpeso10=maxpeso10+peso10(1,:);
minpeso10=minpeso10+peso10(1,:);
maxpeso15=maxpeso15+peso15(1,:);
minpeso15=minpeso15+peso15(1,:);

%8°Dibujamos de la otra manera mostrando el lataeral.
```

```
figure('Position',[85 109 1185 511]);
axis([-400 3600 -100 800]);
grid on
for i=9:1:18
hold on
plot(media_modelo(1,3*i),media_modelo(1,3*i-
1),'blued','MarkerFaceColor','b','MarkerSize',8)
hold on
plot(peso0(1,3*i),peso0(1,3*i-1),'green.','MarkerFaceColor','g','MarkerSize',25)
hold on
plot(peso2(1,3*i),peso2(1,3*i-1),'r.','MarkerFaceColor','r','MarkerSize',25)
hold on
plot(peso4(1,3*i),peso4(1,3*i-1),'cyan.','MarkerFaceColor','c','MarkerSize',25)
hold on
plot(peso6(1,3*i),peso6(1,3*i-
1),'magenta.','MarkerFaceColor','magenta','MarkerSize',25)
hold on
plot(peso8(1,3*i),peso8(1,3*i-
1),'yellow.','MarkerFaceColor','y','MarkerSize',25)
hold on
plot(peso10(1,3*i),peso10(1,3*i-
1),'black.','MarkerFaceColor','black','MarkerSize',25)
hold on
plot(peso15(1,3*i),peso15(1,3*i-
1),'blue.','MarkerFaceColor','blue','MarkerSize',25)
legend('Modelo','Peso 0','Peso 2','Peso 4','Peso 6','Peso 8','Peso 10','Peso
15','Location','Best')
end

%Dibujamos ahora los rangos
for i=9:1:18
hold on
plot(maxmodelo(1,3*i),maxmodelo(1,3*i-1),'bluex-')
hold on
plot(minmodelo(1,3*i),minmodelo(1,3*i-1),'bluex-')
hold on
plot(maxpeso0(1,3*i),maxpeso0(1,3*i-1),'greenx-')
hold on
plot(minpeso0(1,3*i),minpeso0(1,3*i-1),'greenx-')
hold on
plot(maxpeso2(1,3*i),maxpeso2(1,3*i-1),'rx-')
hold on
plot(minpeso2(1,3*i),minpeso2(1,3*i-1),'rx-')
hold on
plot(maxpeso4(1,3*i),maxpeso4(1,3*i-1),'cyanx-')
hold on
plot(minpeso4(1,3*i),minpeso4(1,3*i-1),'cyanx-')
hold on
plot(maxpeso6(1,3*i),maxpeso6(1,3*i-1),'magentax-')
hold on
plot(minpeso6(1,3*i),minpeso6(1,3*i-1),'magentax-')
hold on
plot(maxpeso8(1,3*i),maxpeso8(1,3*i-1),'yellowx-')
hold on
plot(minpeso8(1,3*i),minpeso8(1,3*i-1),'yellowx-')
hold on
plot(maxpeso10(1,3*i),maxpeso10(1,3*i-1),'blackx-')
hold on
plot(minpeso10(1,3*i),minpeso10(1,3*i-1),'--blackx')
hold on
plot(maxpeso15(1,3*i),maxpeso15(1,3*i-1),'--bluex')
```

```
hold on
plot(minpeso15(1,3*i),minpeso15(1,3*i-1),'--bluex')
end

continuar2=questdlg('¿Quitar?','','Ok','Cancelar','Ok');
switch continuar2
    case 'Ok'
    delete(1)
    case 'Cancelar'
end
```

Volver